"Ss. Cyril and Methodius" University in Skopje
FACULTY OF COMPUTER
SCIENCE AND ENGINEERING

Учебник по информатика за седмо оделение

Современ курс за нова дигитална генерација

АВТОРИ:

ЈАСМИН АБАЗИ (225039) – МАВРОВО И РОСТУШЕ, РЕПУБЛИКА МАКЕДОНИЈА

РЕЏЕП МАВМУТИ (221222) – МАВРОВО И РОСТУШЕ,

РЕПУБЛИКА МАКЕДОНИЈА

БРАНКО КИРОВСКИ (225080) – СКОПЈЕ, РЕПУБЛИКА МАКЕДОНИЈА

ВИКТОР КОНЕСКИ (222116) – СКОПЈЕ, РЕПУБЛИКА МАКЕДОНИЈА

Добредојдовте во светот на Информатиката!

Добредојде, дигитален истражувачу!

Оваа книга е направена токму за тебе – љубопитен, креативен и подготвен да дознаеш како функционира светот околу тебе преку технологијата. Во неа ќе научиш како работи компјутерот, што значи програмирање, како да направиш своја презентација или анимација, и – најважно – како да бидеш безбеден и паметен корисник на интернетот.

Информатиката не е само за програмери. Таа е за секого – и за оние што сакаат да креираат, да истражуваат, да играат игри и да решаваат проблеми.

Оваа книга ќе ти покаже дека учењето може да биде забавно, интерактивно и интересно.

Тука нема да наидеш само на текст. Те очекуваат:

- Идеи
- Анимации
- Квизови
- Интересни задачии
- многу предизвици кои ќе ти помогнат да го откриеш твојот диигитален потенцијал

Подготвен(а) си да започнеме?

Ајде да влеземе во светот на Информатиката – свет полн со идеи, кодови, мрежи и многу, многу забава!

Зошто го направивме овој курс?

Во денешниот дигитален свет, децата не се само корисници на технологијата — тие се идни креатори, програмери и иноватори. Со оваа мисла, тргнавме во подобрување на курсот по Информатика за 7мо одделение. Нашата цел не беше само да ги „освежиме“ старите лекции, туку да создадеме целосно ново доживување за учениците — доживување во кое ќе учат преку интеракција, истражување и игра.

Знаевме дека децата на 12-годишна возраст имаат огромна љубопитност, но и краток фокус, па затоа вложивме напори за да го направиме материјалот забавен, визуелно привлечен и јазично разбирлив.

Наместо суви дефиниции, понудивме реални примери, илустрации, анимации и практични задачи што ги поврзуваат апстрактните теми со нивниот секојдневен живот.

Преку овој пристап сакаме да ја разбиеме идејата дека Информатиката е тежок и досаден предмет. Напротив — таа е простор за креативност, логика, дружба со технологијата, но и безбедност во дигиталниот свет.

Веруваме дека кога учениците ќе видат смисла и примена во она што го учат, тие не само што ќе го запомнат материјалот — туку и ќе се вљубат во него.

Овој курс е направен за нив — за љубопитните, креативните и паметните млади умови кои ќе бидат дигиталните лидери на утрешнината.

Цели на курсот

Целта на овој курс е да ти помогне не само да научиш како функционираат компјутерите, туку и да развиеш вештини кои ќе ти бидат корисни во животот, во учењето и во иднината.

Преку интересни примери, интерактивни

активности и практични задачи, ќе ги стекнеш следниве вештини и знаења:

 Разбирање на технологијата околу нас

– Ќе научиш што се компјутери, како работат и од што се составени.

 Користење на алатки за учење и работа

– Ќе работиш со програми за пишување, презентации, табели и ќе создаваш сопствени дигитални проекти.

 Основи на програмирање

– Ќе научиш да размислуваш логички, да решаваш проблеми и да програмираш игри или приказни во Scratch.

 Безбедно користење на интернетот

– Ќе дознаеш како да се заштитиш на интернет, како да комуницираш одговорно и што значи дигитален отпечаток.

 Креативност и изразување преку технологија – Ќе имаш можност да креираш, дизајнираш и претставиш свои идеи користејќи дигитални алатки.

Промоција на курсот

Оваа книга е дел од еден нов, модерен пристап кон Информатиката – пристап што ја прави наставата поинтересна, поврзана со секојдневието и полесна за разбирање. Создадена со љубов, внимание и креативност, книгата е направена со цел да им покаже на учениците дека учењето не мора да биде здодевно — туку може да биде возбудливо, интерактивно и полно со идеи.

Секоја страница е тука за да те инспирира, да ти даде ново знаење и да те охрабри да истражуваш и создаваш.

#Информатика7Разбирливо

Што ќе учиме?

 1. Табели и пресметки со Excel

Ќе научиш:

- како се внесуваат податоци во табела,
- како се прават пресметки,
- како се креираат графици,
- и како табелите ни помагаат во секојдневниот живот.

➡ *Практична тема што ќе ти помогне да се снајдеш со податоци и броеви на лесен начин!*

 2. Решавање логички задачи преку Информатика

Ќе решаваме:

- загатки и логички натпреварувачки задачи,
- задачи што се користат на ИТ-натпревари,
- интересни проблеми кои ќе ти го „вежбаат мозокот“.

➡ *Ќе учиш како да размислуваш како програмер, дури и пред да напишеш една линија код!*

 3. Програмирање со C++

Ќе направиш чекор напред и ќе:

- учиш што е програмски јазик,
- се запознаеш со основни команди, променливи и услови,
- создаваш свои мали програми користејќи C++.

➡ *Програмерскиот свет ќе ти стане поблиску од кога било – сè ќе учиме со јасни примери и објаснувања.*

Овој курс не е само за учење – туку за **размислување, решавање проблеми, истражување и создавање.**
 Подготви се да станеш вистински **дигитален креатор!**

Програма за табеларни пресметки: Функции и можности

Вовед во програмата за табеларни пресметки

🎯 Што ќе научиме во ова поглавје?

Да користиме формули за автоматски пресметки
Да ги украсиме табелите со бои и линии
Да направиме графикони
Да сортираме податоци (од најголем до најмал итн.)

📌 Каде се користи Excel во животот?

- **Наставници** – за евиденција на оценки и успех
- **Библиотекари** – за список на книги
- **Ученици** – за табела со редовност, проектни задачи, имиња
- **Родители** – за трошоци, месечен буџет
- **Секојдневно** – за правење списоци, пресметки, графикони

Слика 1: Икона на Microsoft Excel 2016

📌 ШТО ЌЕ НАУЧИШ ДА НАПРАВИШ?

📌 По оваа лекција, ќе умееш:

- ✍️ Да направиш нова табела
- 📌 Да внесеш текст, броеви и датуми
- 📌 Да направиш формули за пресметки
- 📌 Да форматираш (бои, големина, рамки)
- 📌 Да направиш графикон
- 📌 Да сортираш и филтрираш податоци

📌 Пред да почнеме:

📌 На компјутерот треба да биде инсталирана програмата **Microsoft Excel 2016** (дел од Microsoft Office 2016).

Excel 2016

Што ќе научиме?

- Што е табела

ШТО Е ТАБЕЛА?

Табела = податоци подредени како мрежа.
Има **редови** (хоризонтално) и **колони** (вертикално).
Секоја „кутија“ во мрежата е **ќелија**.

- Колони = букви (A, B, C...)
- Редици = броеви (1, 2, 3...)
- Ќелија = A1, B2, C3...

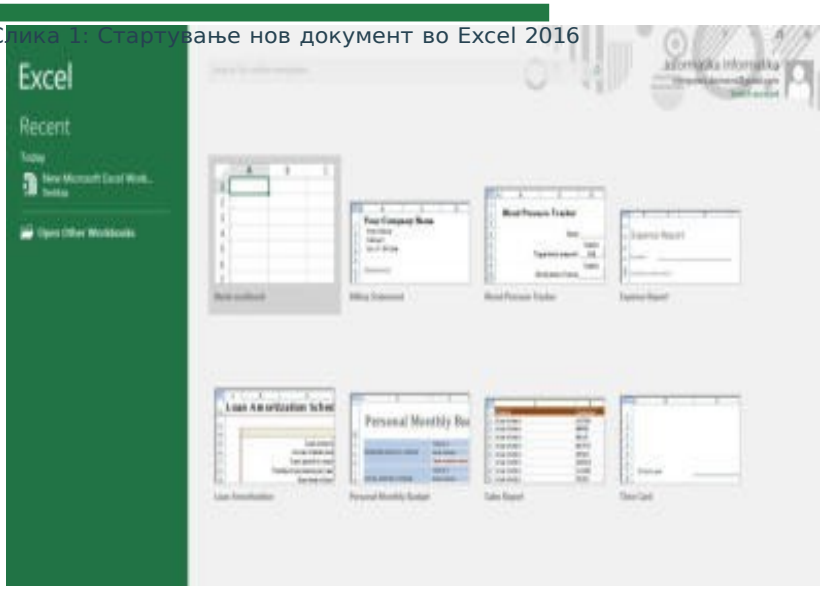
- *Пример: A3 е трет ред во првата колона.*

За да ја отвориме програмата:

1. Кликни на **Start мени**
2. Напиши **Excel**
3. Избери иконата

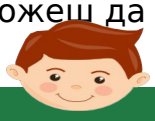
Притоа се појавува прозорецот :

Слика 1: Стартување нов документ во Excel 2016



Кога ќе се вклучи Excel, ќе го видиш:

- **Blank Workbook** (празна табела за почеток)
- **Recent Documents** – претходно отворени датотеки
- Можеш да избираш и шаблони (готови табели)



Што е работна тетратка (Workbook)?

Тоа е целиот Excel документ.
Се зачувува со наставка **.xlsx**
Пример: oceni.xlsx

Што е работен лист (Worksheet)?

Секој документ (тетратка) содржи **листани страници**, наречени **лисови (Sheet1, Sheet2, ...)**
→ Секоја страница може да има своја табела.

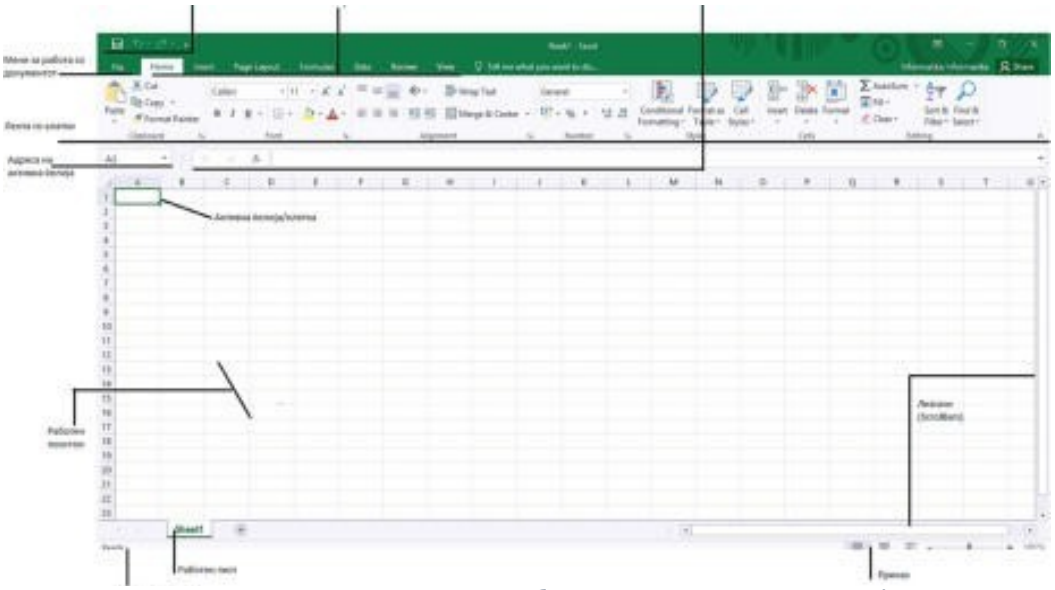
Што е работен лист (Worksheet)?

Секој документ (тетратка) содржи **листани страници**, наречени **лисови (Sheet1, Sheet2, ...)**
→ Секоја страница може да има своја табела.

Што е табела?

Табела е мрежа составена од колони и редици, слично како табелата на училишната табла.

модифицира според потребите на корисникот.



Слика 2: Работен прозорец на MS Excel 2016

2. **Лента со менија (Menu Bar)** – е мени-лентата која претставува групирање на командите според нивните задачи, всушност, лента со наредби за соодветна картичка.

3. **Лента со алатки** - се појавува или менува зависно кое мени е избрано.

4.**Адреса на активна ќелија (Name Box)** – претставува поле за приказ на името на ќелијата која е селектирана, односно ја прикажува активната ќелија.

5. **Formula Bar** – е лента за внесување формули и функции, приказ на податоците или содржината на активната ќелија.

6. **Активна ќелија** – е ќелијата во која корисникот внесува податоци.

7. **Sheet Tabs** – е лента со работни листови.

8. **Scroll Bar** – се ленти за движење низ работниот прозорец: хоризонтална и вертикална.

9. **Статусна лента (Status Bar)** – се наоѓа на долниот дел од прозорецот и истата може да се прилагодува за приказ на повеќе опции, од кои повеќето му даваат на корисникот информации за тековниот лист.

10. **Мени File** – ги содржи основните наредби за работа со документот во Excel.

1.1.1 Работа со документи во Excel

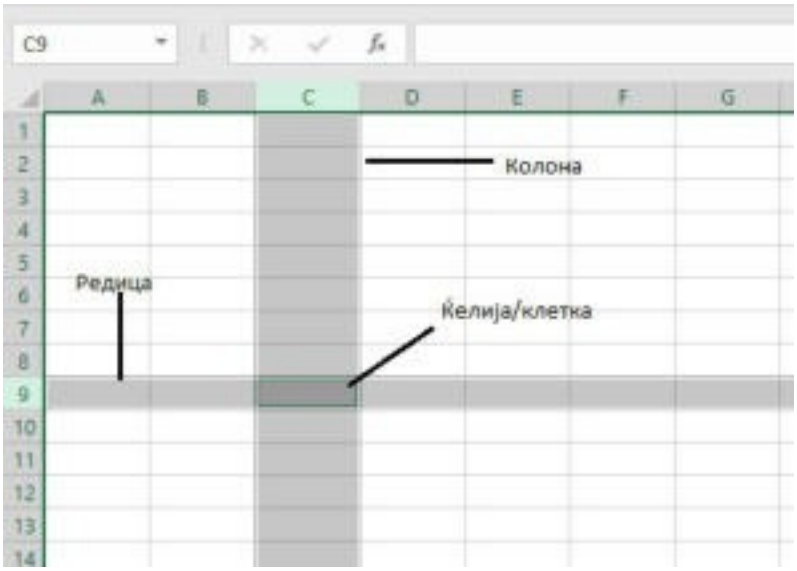
Ајде да се потсетиме:

Менито File во Excel функционира исто како и во Word или PowerPoint. Таму ги наоѓаме основните команди за управување со документите.

Операција	Што прави?	Пример / Совет
New	Отвара нов празен документ	Почни табела за нов проект
Open	Отвора постоечки документ	Пример: `Оцени.xlsx`
Save	Го зачувува тековниот документ	Секогаш зачувај често!
Save As	Зачувува под ново име или локација	`oceni_април.xlsx`
Print	Го печати документот	Распоред на часови
Share	Испраќа преку е-пошта или во тим	За групни проекти
Export	Сменува формат (на пр. во PDF)	Испрати табела како PDF
Close	Го затвора документот	Завршено за денес ✅

1.1.3 *Работа со основните елементи на работниот лист*

На следниот прозорец прикажани се основните елементи на работниот лист: табела, редица, колона и ќелија:



Слика 3: Основни елементи на работниот прозорец

Задача

Вежба во Excel:
Отвори нов лист
Во ќелија A1 напиши Име
Во A2, A3, A4 напиши 3 имиња
Селектирај ги сите имиња со **Shift**
Селектирај само A2 и A4 со **Ctrl**

Сè започнува со една ќелија...

🔹 Што е ќелија?

- Место каде внесуваме податоци (текст, број, формула...)
- Секоја ќелија има име (A1, B3, D5...)
- Ќелијата во која работиме во моментот се вика 🔹 **активна ќелија**

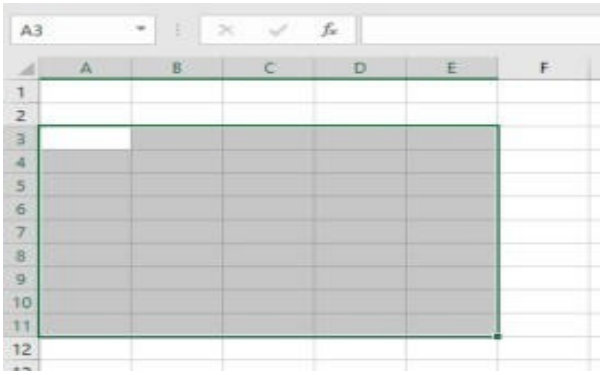
🔹 ШТО Е РАНГ НА ЌЕЛИИ?

🔹 **Ранг на ќелии** значи **група од соседни ќелии** што се една до друга – **во форма на правоаголник**.

🔹 Како изгледа еден ранг?

➡ Пример: A3:E11
🔹 Тоа значи:

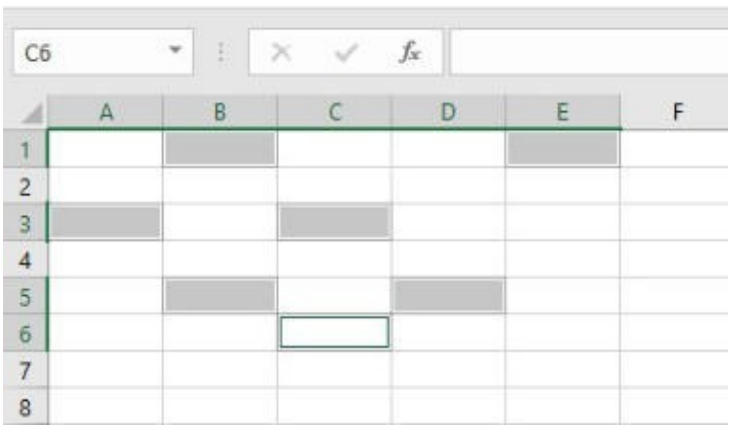
- Започнува од ќелијата **A3**
- Завршува со ќелијата **E11**
- Се опфаќаат сите ќелии **од A до E** и **од редица 3 до 11**



Слика 4: Селектирање последователни/сосоедни ќелии

НЕСОСЕДНИ ЌЕЛИИ (неповрзани ќелии)

🔹 Понекогаш, не сакаме да селектираме ќелии една до друга — туку **одвоени**, на различни позиции.
🔹 Тогаш користиме тастерот **Ctrl**.



✍ **Како се селектираат?**

1. Кликни на првата ќелија (на пр. **A1**)
2. Држи притиснато **Ctrl**
3. Кликни на другите ќелии кои сакаш да ги избереш (на пр. **C3**, **E5**, **F10**)
4. Сите тие ќелии ќе останат обележани

Слика 5: Селектирање несоседни ќелии

Копче од тастатура	Движење
Enter или ↓	Една ќелија подолу
↑	Една ќелија погоре
←	Една ќелија налево лево
→	Една ќелија надесно десно
Ctrl + ←	Првата ќелија во редицата
Ctrl + →	Последната ќелија во редицата
Home	Почеток на редот
End	Крај на редот
Ctrl + Home	Почеток на табелата
Ctrl + End	Крај на табелата

За движење низ работниот документ ќе ги употребуваме лентите за движење (Scrool Bars) кои се наоѓаат на десната страна и во долниот дел на работниот прозорец, како и копчињата од тастатурата:

Табела 2: Движење низ работниот прозорец

Вежба 1

Во документот од вежба 1, во работниот лист Sheet1 да се селектираат ќелиите B4:F8 и со употреба на алатката Fill Color, од менито Home, да се обојат со сина боја. На работниот лист дајте му име: соседни. Промените кои се направени во документот да се зачуваат.

Во истиот документ додадете нов работен лист. Именувајте го работниот лист во несоседни. Селектирајте ги ќелиите: A2, C2, E2, G2, B3, D3, F3, A4, C4, E4, G4, B5, D5, F5, A6, C6, E6, G6, B7, D7, F7. Промените кои се направени да се зачуваат.

Упатство

Дел 1

- Hint 1: Во кој лист се започнува оваа вежба?
- Hint 2: Размисли: како можеш да селектираш група од ќелии што се една до друга?
- Hint 3: Кој тастер можеш да го користиш за тоа?
- Hint 4**:** Каде се

Провери си го знаењето!

1. Зошто служи програмата Excel 2016?
2. Како се викаат документите кои се креирани во програмата за табеларни пресметки?
3. Кои се елементите на работниот прозорец на програмата за табеларни пресметки?
4. Како се означени хоризонталните делови во активниот документ во програмата за табеларни пресметки?
5. Како се вика пресекот на колона и редица

1.2 Работа со податоци во табела

Цел: Да научиш како да внесуваш податоци (текст и бројки) во табела

Ситуација:

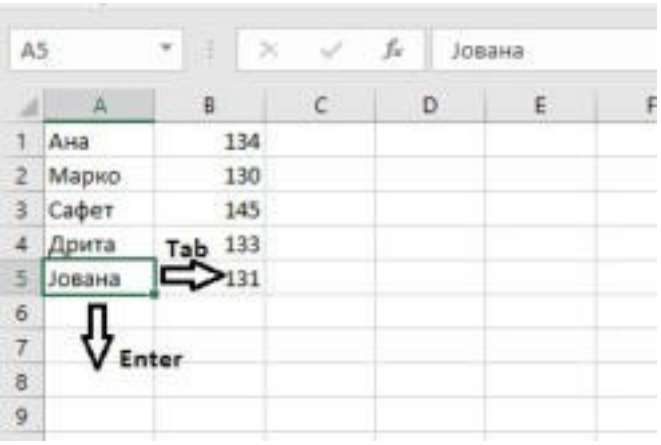
Сакаш да направиш табела со **имиња на ученици и нивната висина**. За тоа ќе ги користиш ќелиите во Excel.

Што треба да направиш?

Податоците секогаш се внесуваат во **активната ќелија**
Потоа се движиш со **Enter**, **Tab** и **стрелки од тастатурата**

✍ Чекори претставени како hints:

1: Каде треба прво да кликнеш за да започнеш со внесување?
2: Кој тастер те носи *во следната редица* по внесување на име?
3: Кога ќе завршиш со внес на имиња, како ќе се префрлиш во следната колона за висина?
4: Кои тастери можеш да ги користиш за да се движиш по табелата?
5: Како ќе го зачуваш фајлот и под кое име?



Слика 1: Внесување податоци во табела

Каде се прикажуваат податоците во Excel?

Забелешка од вежбата:

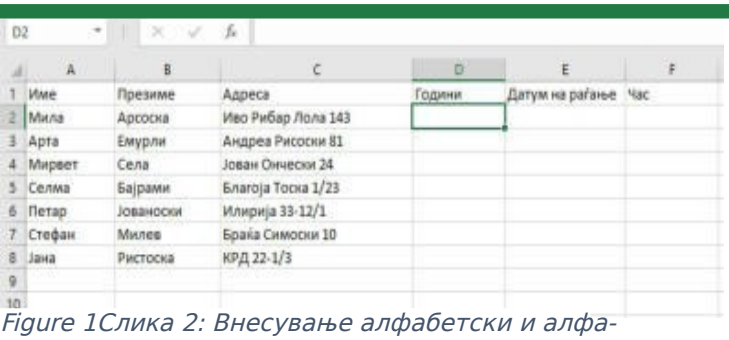
Кога внесуваш податок во табела:

- ✏ Тој се појавува и во **Formula Bar** (лентата над табелата)
- Адресата на ќелијата (на пример B3) се прикажува во **Name Box** (лево од Formula Bar)

Пример:

Внесуваш зборот Јована во

1.1.1 Типови податоци



	A	B	C	D	E	F
1	Име	Презиме	Адреса	Години	Датум на раѓање	Час
2	Мила	Арсоска	Иво Рибар Лола 143			
3	Арта	Емурли	Андреа Рисоски 81			
4	Мирвет	Села	Јован Ончески 24			
5	Селма	Бајрами	Благоја Тоска 1/23			
6	Петар	Јованоски	Илирија 33-12/1			
7	Стефан	Милев	Браќа Симоски 10			
8	Јана	Ристоска	КРД 22-1/3			
9						
10						

Figure 1Слика 2: Внесување алфабетски и алфа-нумерички податоци

Во табела можеш да внесуваш три типа на податоци:

Тип на податок	Што содржи?	Пример
Алфабетски	Само букви (текст)	`Ана` , `Скопје`
Нумерички	Само бројки	`15` , `2024` , `7.5`
Алфа-нумерички	Комбинација од букви и бројки	`улица 11` , `ученик3`

Нумерички податоци

Што значи „нумерички“?

Нумерички податоци се **бројки**. Но не се само едноставни броеви! Во Excel, тоа можат да бидат:

Тип на бројка	Пример
Цел број	`5` , `123` , `-8`
Децимален број	`7.5` , `0.99`
Датум	`14.03.2012`
Час	`12:45` , `08:00`
Валута	`$15` , `€1200`

 Работен фајл: Vezba2.xlsx

 Hint 1: Додади нов работен лист – знаеш ли каде е „+“ копчето?
 Hint 2: Преименувај го листот во Податоци
 Hint 3: Во првиот ред (A1 до G1) внеси наслов за секоја

	A	B	C	D	E	F	G	H
1	Име	Презиме	Адреса	Години	Датум на раѓање	Час		
2	Мила	Арсоска	Иво Рибар Лола 143	13	15.03.2006	18:30		
3	Арта	Емурли	Андреа Рисоски 81	14	18.02.2005	22:40		
4	Мирвет	Села	Јован Ончески 24	13	22.08.2006	14:10		
5	Селма	Бајрами	Благоја Тоска 1/23	12	29.09.2007	19:20		
6	Петар	Јованоски	Илирија 33-12/1	15	12.12.2004	17:22		
7	Стефан	Милев	Браќа Симоски 10	16	28.03.2003	03:15		
8	Јана	Ристоска	КРД 22-1/3	14	03.03.2005	06:25		
9								

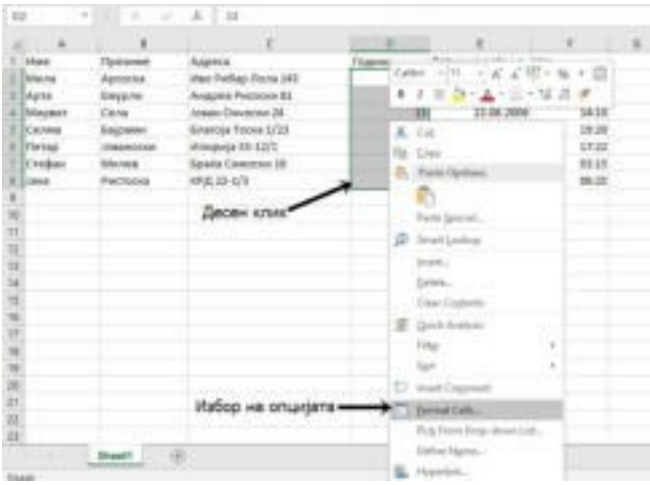
Слика 3: Внесување нумерички податоци

1.1.1 Формат на бројни (нумерички) податоци

Како да го промениш форматот на бројка?

 Hint 1: Селектирај ќелии со бројки
 Hint 2: Десен клик → избери **Format Cells**
 Hint 3: Во прозорецот што ќе се појави, избери што значи таа бројка:

- Number (обичен број)
- Currency (пари)
- Date (датум)
- Time (час)
- Percentage (проценти)

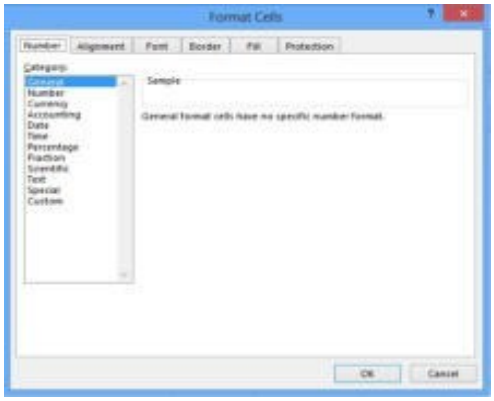


Слика 4: Избор на опцијата Format Cells

Формати на бројни податоци во Excel

Кога ќе го отвориш прозорецот Format Cells, ќе видиш табот Number.

Таму се наоѓаат сите формати што Excel ги користи за бројки.



Слика 5: Прозорецот на Format Cells

📋 Табела: Формати и нивно значење

Категорија	Што прави?	Пример
General	Без специфичен формат	`1234`
Number	Обични бројки (со децимали и +/-)	`-55.20`
Currency	Пари со симбол (€, $, ден.)	`€200.00`
Accounting	Пари со уредено порамнување	`ден 1200.00`
Date	Датум	`15.06.2025`
Time	Час	`13:45:00`
Percentage	Приказ во %	`0.75 → 75%`
Fraction	Дропки	`1/2`, `3 3/4`
Scientific	Експоненцијално	`1.2E+03`
Text	Бројот се чита како текст	`01234` (без губење на нула)
Special	Посебни формати (на пример телефонски броеви)	`070-123-456`
Custom	Креираш свој формат	`ДД-ММ-ГГГГ`, `## ден.`

Провери си го знаењето!

1. Во која ќелија се внесува податок?
2. Кое копче или копчиња се кликнуваат по внесување на податок во ќелијата?
3. Набројте различни формати на бројни податоци!

1.1 Уредување табела

🎯 Цел: Да научиме како да смениме податоци во табела – без да ја избришеме целата

❓ Кога се прават грешки...

Во реалноста често:

- ❌ Го внесуваме погрешно името
- ❌ Забораваме колона (на пр. датум на раѓање)
- ❌ Сакаме да додадеме или поправиме нешто

Не треба да почнуваш од почеток – само **ја уредуваш табелата.**

✍️ 1.3.1 Промена на податоци

🔹 Пример 1: Целосна замена на вредност

🖱 **Hint:** Кликни на ќелијата (нпр. B4), и само напиши нов текст.
🔄 Статусната лента ќе покаже: **Ready**

📌 Пример: Наместо Села, напиши Селими

🔹 Пример 2: Додавање/поправка на буква

🖱 **Hint:** Двапати кликни внатре во ќелијата (нпр. B2)
🔄 Курсорот се појавува внатре
🔄 Статусната лента ќе покаже: **Edit**

📌 Пример: Сеими ➜ Севими (вметнуваш „в“)

Бришење на податоци

🗑 Селектирај ќелии → притисни **Delete** на тастатурата

 Пример: Сакаш да избришеш грешна година, селектирај ја ќелијата и притисни Delete.

1.1.1 Активности со колони и редици

Како се додава нова колона?

 Колони = вертикални (A, B, C, D…)

 Hint чекори:
1 Кликни на буквата на колоната ПРЕД која сакаш да додадеш нова
2 Десен клик
3 Избери **Insert**

 Пример: Сакаш нова колона помеѓу B и C → означи C → десен клик → Insert

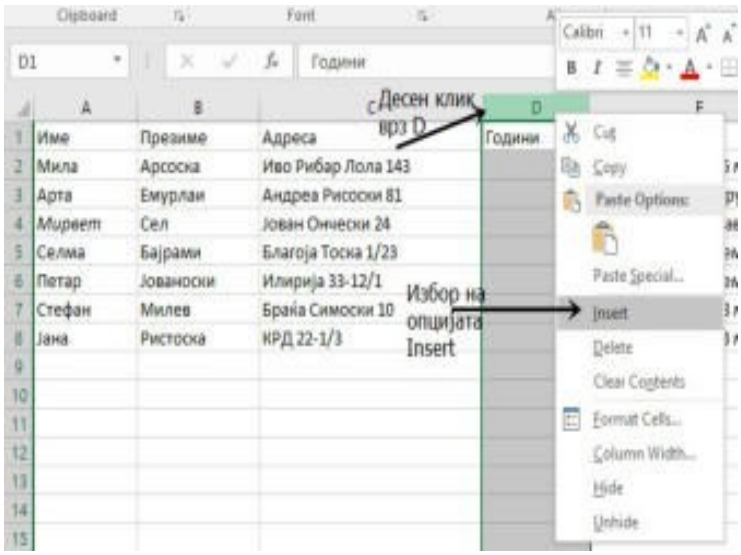
---

Како се додава нова редица?

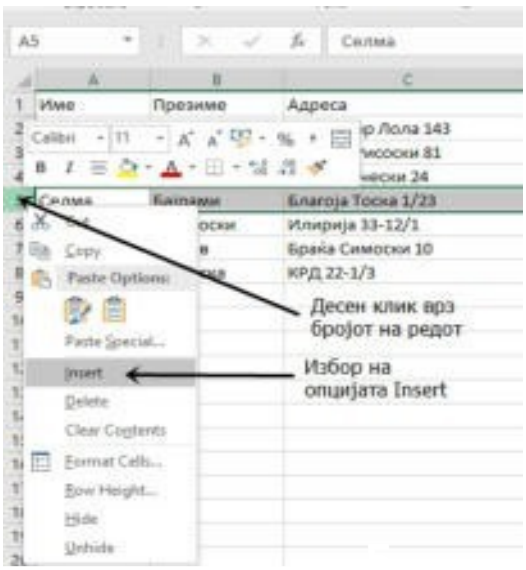
 Редици = хоризонтални (1, 2, 3, 4…)

 Hint чекори:
1 Кликни на бројот на редот ПОД кој сакаш да додадеш нов
2 Десен клик
3 Избери **Insert**

 Пример: Сакаш нова редица помеѓу 4 и 5 → означи редица 5 → десен клик → Insert



Слика 3: Вметнување колона во табела



Слика 4: Вметнување редица во табела

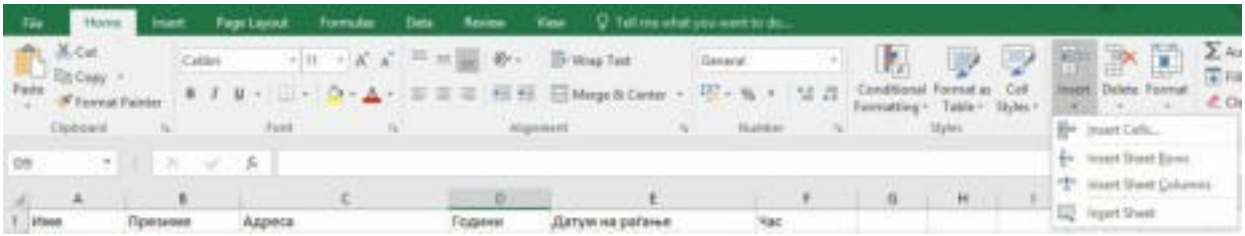
Алтернативен начин:

Можеш да додадеш нова колона/редица преку **менито Home** во горниот дел на Excel.

Hint-чекори:

1 Селектирај редица или колона
2 Од горниот дел на екранот избери Home
3 Во делот **Cells**, кликни на **Insert**
4 Избери:

- Insert Sheet Rows → додава редица
- Insert Sheet Columns → додава колона



Слика 5: Вметнување колона и редица преку менито

1.1.2 Менување на димензиите на колоните и редовите

Цел: Да научиме како да направиме колоната поширока или редот повисок – за податоците да се гледаат убаво и јасно.

Кога ни треба ова?

- Текстот не се гледа целосно?
- Името на ученик е отсечено?
- Бројката не собира во ќелијата?

Решението: **Ја прошируваш колоната** или **ја зголемуваш висината на редот**!

Како да го направиш тоа?

Hint-чекори:

За колона:

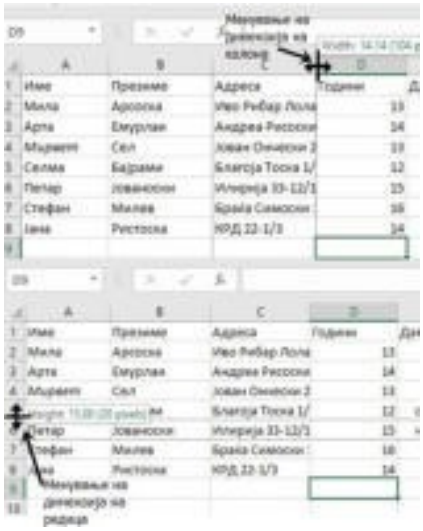
1 Стави го покажувачот **помеѓу две букви**, на пример помеѓу B и C
2 Курсорот ќе се смени (⇔)
3 Притисни и **влечи лево/десно**
 Колоната се проширува или стеснува

За редица:

1 Стави го покажувачот **помеѓу два броја**, на пример помеѓу 3 и 4
2 Курсорот ќе се смени (⇕)
3 Притисни и **влечи нагоре/надолу**

⚡ **Брз трик:**

 Двоен клик на линијата до буквата или бројот ➜ Excel автоматски ќе ја прилагоди ширината или висината според содржината!



Слика 6: Менување димензија на колона и редица

Задача

Отворете го документот „Vezba2.xlsx“ и изменете ги димензиите на колоната каде што се наоѓа датумот на раѓање и насловната редица. Потоа, извршете ја постапката за менување димензии на колона/ред преку опцијата Format од менито Home, со цел да добиете исти димензии за сите колони и редици.

Запомни!

 Едитирањето значи **поправање или дополнување** на веќе внесени податоци.

➡ Можеш да уредиш податок:

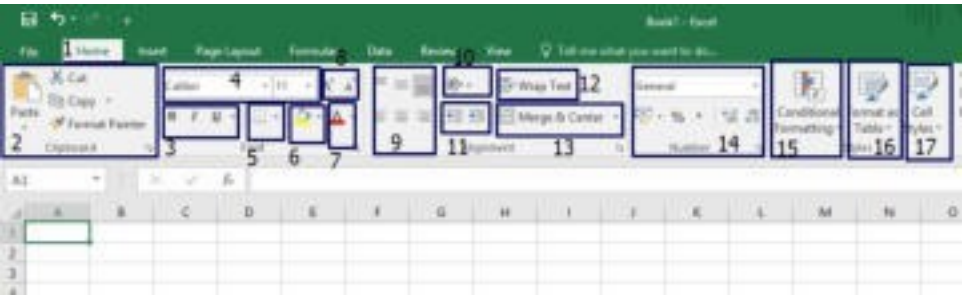
- со **клик** на ќелија и директно нов внес (статусна лента: Ready)

Ш

Форм **но да с**

Што можеме да форматираме?

Ќелии, редици, колони и цела табела – сè може да се уреди!



1. Менито Home.
2. Наредбите за копирање, отсекување, вметнување и алатака за форматирање.
3. Стиловите: задебелен, накосен и подвлечен.
4. Избор на фонт и големина на фонт.
5. Избор на рабови.
6. Избор на боја на ќелиите.
7. Избор на боја на фонтот.
8. Алатки за зголемување/намалување на буквите. 9.Ориентација на податоците.
10. Вовлекување.
11. Разделување на текстот
12. Спојување на ќелии и подредување на центар.
13. Избор на тип на податок.
14. Условно форматирање.
15. Форматирање на табела.
16. Стилови на ќелија.

Освен брзите алатки во **менито Home**, Excel нуди и **напредно форматирање** преку:

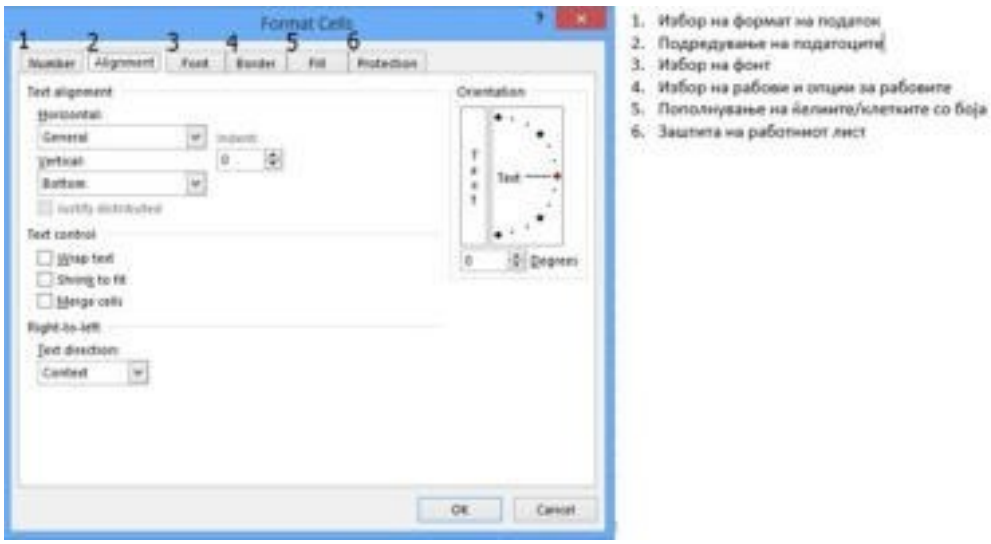
Десен клик → Format Cells...

 Ќе ти се појави прозорец со повеќе табови (страници). Еве ги најкорисните:

Alignment (Подредување)

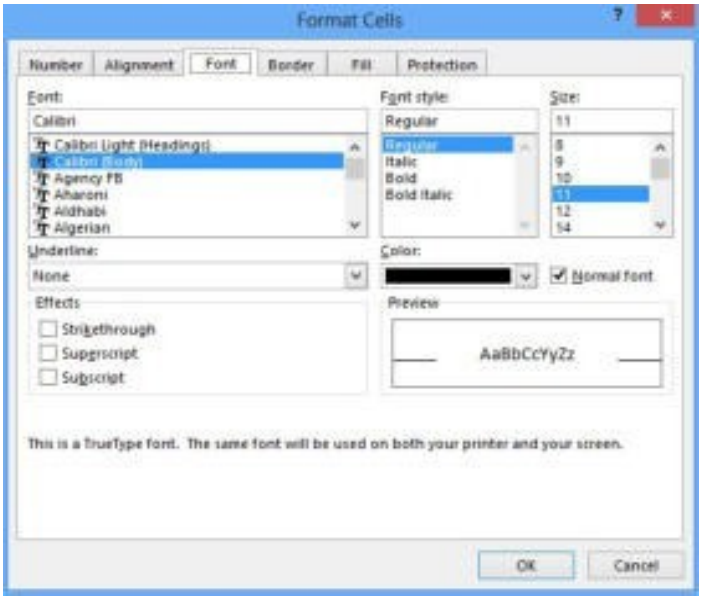
 Овозможува:

- **Horizontal / Vertical Align** – дали текстот ќе биде лево, десно, горе, долу или центрирано.
- **Text Orientation** – текстот под агол или вертикално.
- **Wrap Text** – текстот се распоредува во повеќе реда во истата ќелија.
- **Shrink to Fit** – текстот автоматски се смалува за да собере.
- **Merge cells** – спојува повеќе ќелии во една.


1. Избор на формат на податок
2. Подредување на податоците
3. Избор на фонт
4. Избор на рабови и опции за рабовите
5. Пополнување на ќелиите/клетките со боја
6. Заштита на работниот лист

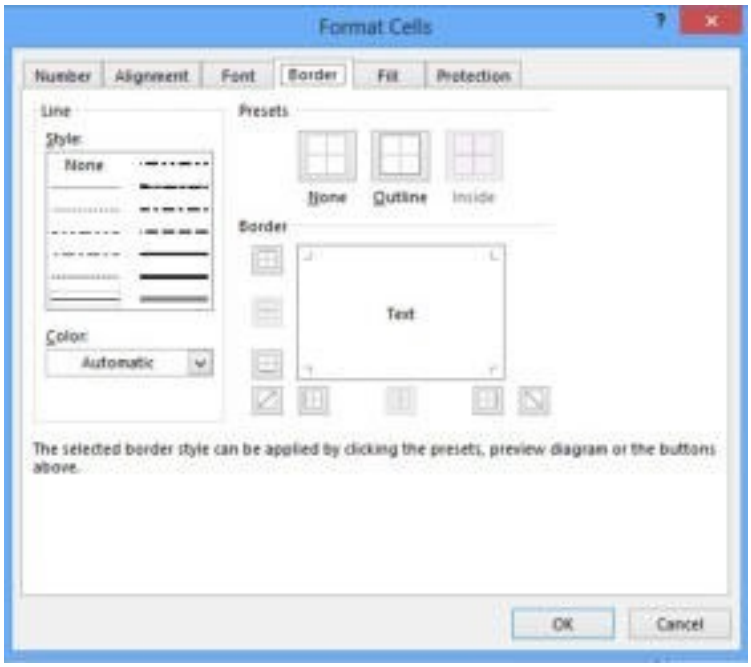
Слика 2: Форматирање на податоците преку менито Format Cells

Преку избор на табот **Font**, се избира облик на пишување на букви, стилови, големина и останатите ефекти кои се однесуваат на фонтот:



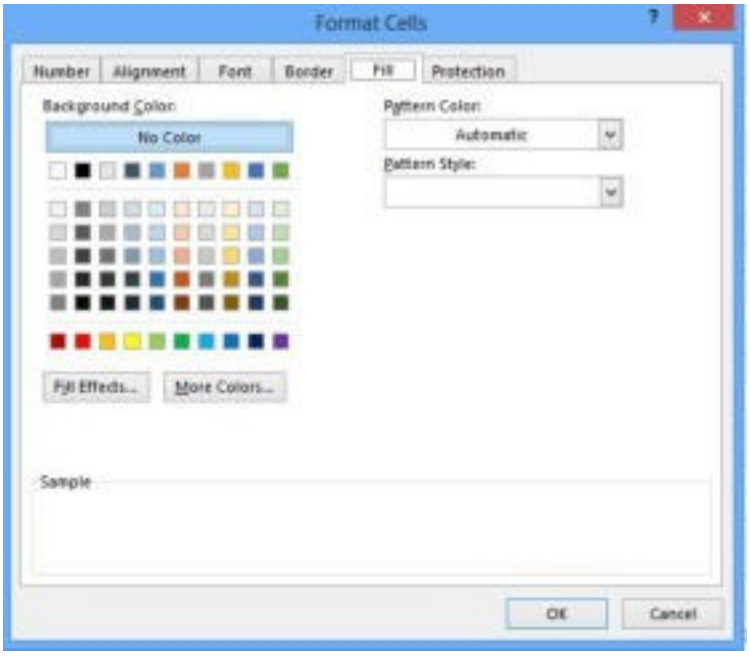
Слика 3: Избор на алатките за фонт

Во опцијата Border, се избира стил, дебелина и боја на рабовите на ќелиите на табелата:



Слика 3: Избор на алатките за фонт.

Преку опцијата Fill се избира боја за пополнување на ќелиите:



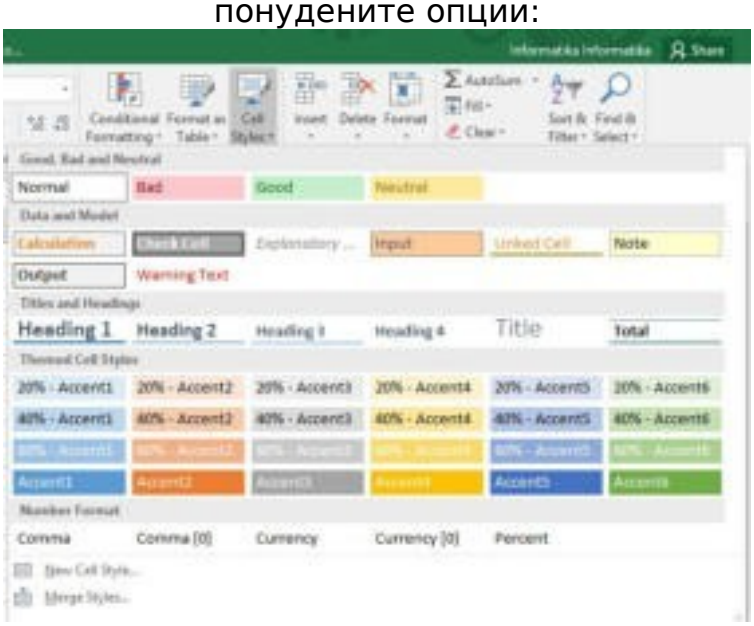
Слика 5: Избор на боја на подлогата на ќелијата

Задача

Отворете нов работен документ и во првиот работен лист (Sheet1) креирајте распоред на часови:

- во ќелиите A1, B1, C1, D1, E1 внесете ги имињата на деновите во неделата во кои посетувате настава;
- во ќелиите кои се наоѓаат под насловните ќелии внесете ги предметите согласно со распоредот на часови;
- додадете нова колона пред колоната понеделник и внесете реден број на часовите;
- додадете нов ред над насловните ќелии;
- спојте ги првите шест ќелии и напишете наслов „Распоред на часови“;
- насловните ќелии во кои се напишани имињата на деновите дајте им фонт Calibri, големина четиринаесет (14), стилови задебелени и накосени, боја на буквите темносина, а боја на позадината на ќелиите светлосина;
- на целата табела додадете рабови;
- димензиите на колоните/редиците поставете ги по потреба;

Форматирањето на ќелиите со готов формат може да се избере и преку избор на алатката Cells Styles од менито Home, како што е прикажано на сликата. Но, потребно е претходно ќелиите да бидат селектирани за да биде имплементиран стил кој корисникот го избрал од понудените опции:



Слика 6: Избор на стилови на ќелии

Покрај готовите формати на ќелии, корисникот може да избере готов формат на табели преку избор на алатката Format As Table, од менито Home, кој е прикажан на следната слика:

Провери си го знаењето!

1. Што е форматирање?
2. Која е разликата меѓу форматирање и уредување табела?
3. Кои алатаки најчесто се употребуваат за форматирање табела?
4. Како се уредуваат порамнувањето и ориентацијата на текстот во ќелиите?
5. Кои опции можеме да ги избереме преку наредбата Format Cells? Дали можеме да пристапиме до нив на друг начин? Објасни!
6. Која е постапката за разделување текст?
7. Зошто служи наредбата Merge Cells?
8. Зошто служи опцијата Cells Styles од менито Home?
9. Дали употребувате готов формат на табели?

Задача

Додадете нов работен лист и креирајте табела во која ќе внесете

помеѓу часовите. Форматирајте ја табелата со примена на стилови на ќелии (**Cell Styles**) од менито **Home** или избор на готов формат на табела преку опцијата **Format Table** повторно од менито **Home**.

1.5 Автоматско пополнување податоци во табелата

Зошто да внесуваш рачно, кога можеш автоматски?

Замисли треба да внесеш броеви од 1 до 100 во табела... 😩
Можеш еден по еден, ама тоа ќе ти одземе многу време.

Токму тука ни помага **Autofill** – една од најкорисните алатки во Excel!

🔹 Што е Autofill?

Autofill ти овозможува:

- брзо да продолжиш **редослед од броеви**
- автоматски да пополниш **датуми, денови, месеци**
- да **повториш вредности** што често ги користиш
- да го **копираш истиот податок** во повеќе ќелии
- да **продолжиш формули** без да ги пишуваш пак

🔹 Примери што можеш да ги направиш со Autofill:

- 1, 2, 3, 4...
- Јануари, Февруари, Март...
- Понеделник, Вторник...
- 5, 10, 15, 20 (ако ги поставиш првите две ќелии)

Како се користи Autofill?

1. Внеси почетни вредности (на пример: 1 и 2)
2. Селектирај ги тие ќелии
3. Фати го **долниот десен агол** од селекцијата (ќе се појави мал црн плус)
4. Повлечи надолу или настрана – и Excel ќе ги пополни останатите автоматски!

Покажувач	Значење
+	Автоматско пополнување ќелии
	Копирање содржина на ќелија
	Преместување содржина на ќелија
	Менување димензија на колона
	Менување димензија на редица
	Означување колона
	Означување редица
	Означување ќелија

Табела 1: Форми на покажувачот

Што треба да направиш?

 Отвори **нов документ во Excel 2016**

➤ Колона A – Парни броеви

 Почни од бројот **2**, продолжи до **22**.

 Hint: Напиши првите два броја (2 и 4), селектирај ги, и повлечи надолу со

➤ Колона B – Непарни броеви

 Почни од бројот **3**, продолжи до **23**.

 Hint: Напиши 3 и 5. Повлечи го Autofill.

➤ Колона C – Името Ана

 Пополни повеќе ќелии со истото име **Ана**.

 Hint: Внеси Ана во првата ќелија и користи Autofill

Зошто користиме формули и функции во Excel?

Excel не е само за правење убави табели — **тој е и калкулатор со супермоќи!**
Со помош на **формули и функции**, можеме да направиме брзи и точни пресметки без да користиме дигитрон.

Главната цел е:

Брзи и прецизни пресметки со само неколку клика!

Што се формули во Excel?

Формулите се пресметки што ги пишуваме во ќелиите за да добиеме резултат.
Тие почнуваат со **еднакво (=)** и може да користат **броеви, ќелии и математички знаци**.

Знак	Математичка операција
+	Собирање
-	Одземање
*	Множење
/	Делење
^	Степен/експонент

Табела 1: Математички оператори за вршење пресметки

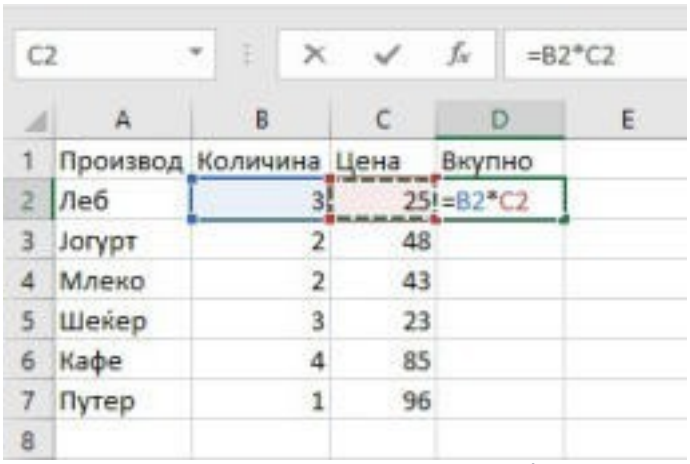
Забелешка!

вистина (True) или невистина (False). При креирање на формулите може да се користат и загради.

🛒 Пример од продавница: Како Excel пресметува колку треба да платиме?
Замисли си дека си во супермаркет и имаш листа со производи. Сакаш да знаеш **колку ќе те чини сè заедно**.
Еве како Excel може да ти помогне!

🧮 Чекори за пресметка со формула:

1. 🖱️ Оди во ќелијата каде што сакаш да го добиеш резултатот — на пример D2.
2. ⌨️ Напиши =, за да му кажеш на Excel дека правиш пресметка.
3. 🖱️ Кликни на ќелијата каде што пишува **количината** (на пример B2).
4. ✖ Стави * за множење.
5. 🖱️ Кликни на ќелијата каде што пишува **цената по парче** (на пример C2).
6. ⏎ Притисни **Enter** – Excel ќе го пресмета вкупниот износ.



Слика 1 : Пресметка со примена на формула

1.1.1 Функции

2 Во Excel, **функциите се како магични формули** – готови се, ти само треба да им дадеш што да пресметаат!

3 ✍ Формата на функција изгледа вака:

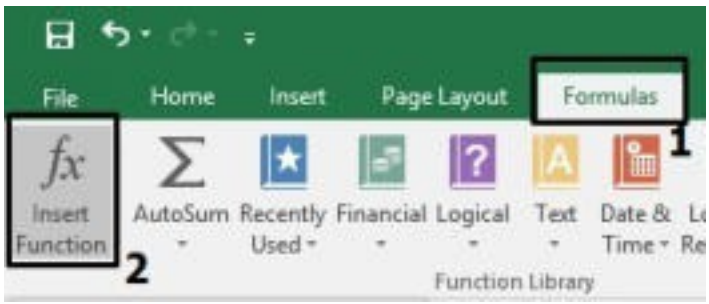
4 CopyEdit
5 =ИМЕ_НА_ФУНКЦИЈА(аргументи)

6 🔹 *Аргументите* се ќелии или опсези (региони) од ќелии кои ги користиш во пресметката.

Најчесто користени функции се:

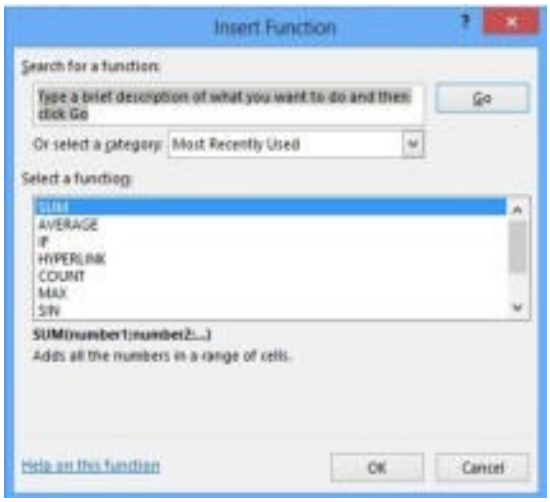
SUM - пресметува збир на бројни вредности во група ќелии.
MIN - дава најмала бројна вредност од група на ќелии.
MAX - дава најголема бројна вредност од група на ќелии.
AVERAGE - пресметува средна вредност од бројните вредности во група ќелии.
COUNT – го дава бројот на нумерички внесови во селектираните ќелии.
COUNTIF- го дава бројот на внесови во селектираните ќелии доколку е задоволен определен услов.

Постапката за внесување на функција започнува со селектирање на ќелијата во која ќе се изврши пресметката, а потоа преку менито **Formulas** и наредбата **Insert Function** пристапуваме кон внесувањето на функцијата.

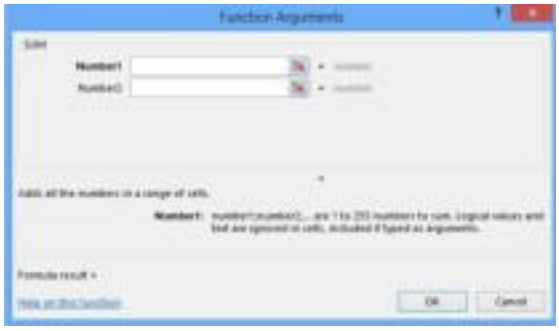


Слика 4: Постапка за внесување функции

Притоа, се појавува следната слика од која избираме категорија и ја избираме функцијата:



За да се изврши пресметката со помош на избраната функција додаваме и аргументи од следниот прозорец:



Со кликнување на копчето **OK** од прозорецот ќе го добиеме резултатот од пресметката.

Но, понекогаш во ќелијата во која треба да се прикаже резултатот од формулата или функцијата се прикажуваат симболите за грешка :#####, #DIV/0!, #NAME?, #REF!, #VALUE!. Тие уште се викаат и пораки за грешка во формулата

Пораки за грешки	Значење
#####	Колоната е тесна да го прикаже резултатот
#DIV/0!	Делење со нула не е можно
#NAME?	во формулата има погрешно запишана адреса на ќелија
#REF!	Во формулата има адреса на ќелија која е избришана
#VALUE!	во формулта една од ќелиите содржи текст

или функцијата (**Error message**)

Задача

е кре
едната слика:

	A	B	C	D	E	F	G
1		Број на прочитани книги по месеци					
2	Име и презиме на ученик	Јануари	Февруари	Март	Април	Мај	Вкупно прочитани книги
3	Ана Михајлоска	4	3	2	1	0	
4	Сара Етеми	5	2	2	1	1	
5	Арта Тахири	4	3	1	2	1	
6	Јован Спироски	3	1	2	1	1	
7	Дритон Емини	6	1	3	2	0	
8							
9	Вкупно прочитани книги по месеци						
10	Најмногу прочитани книги						
11	Најмалку прочитани книги						
12	Просечна вредност на прочитани книги						
13	Вкупно податоци						
14	Колку ученици НЕ прочитале ниту една книга?						

Провери го своето знаење

1. Што се формули, а што се функции?
2. Опишете ја разликата меѓу формулите и функциите!
3. Кои се најчесто употребувани функции?
4. Истражете, набројте и наведете други функции на Excel, освен изучуваните во наставната единица!
5. Напиши ја синтаксата на функцијата која пресметува збир на неколку

1.1 Сортирање податоци

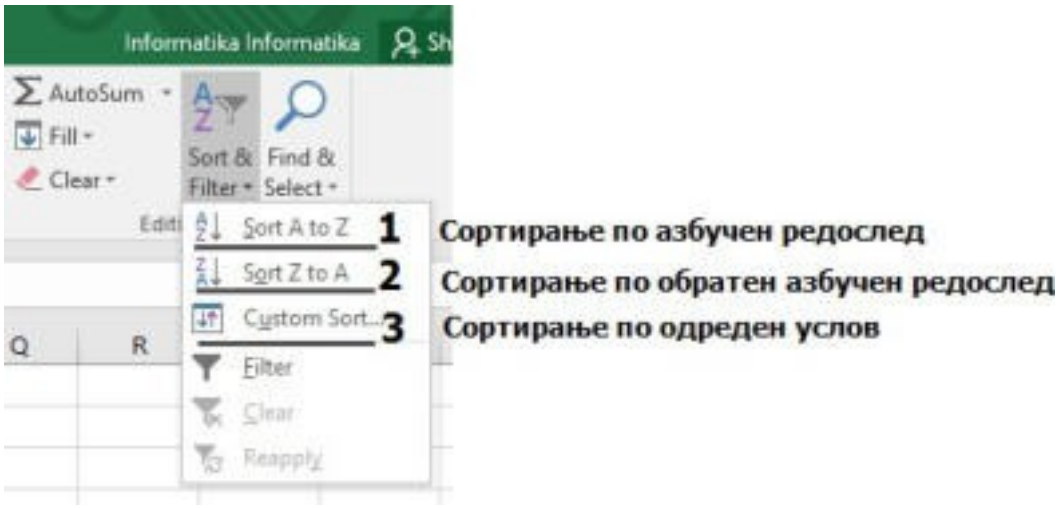
 Кога имаш долга табела, тешко се наоѓаат податоците. Excel има „магичен алат“ кој ти овозможува **да ги подредиш работите онака како што сакаш!** Тоа се вика **сортирање**.

Што е тоа сортирање?

✏ *Сортирање значи да ги подредиш редовите во табелата по одредено правило.*

На пример, можеш да ги подредиш:

- По **азбучен редослед** (А–Ш или обратно)
- По **бројки** (најмало до најголемо или најголемо до најмало)
- По **датум** (најстар до најнов и обратно)
- Или дури и **по бои**, ако имаш условно форматирање!

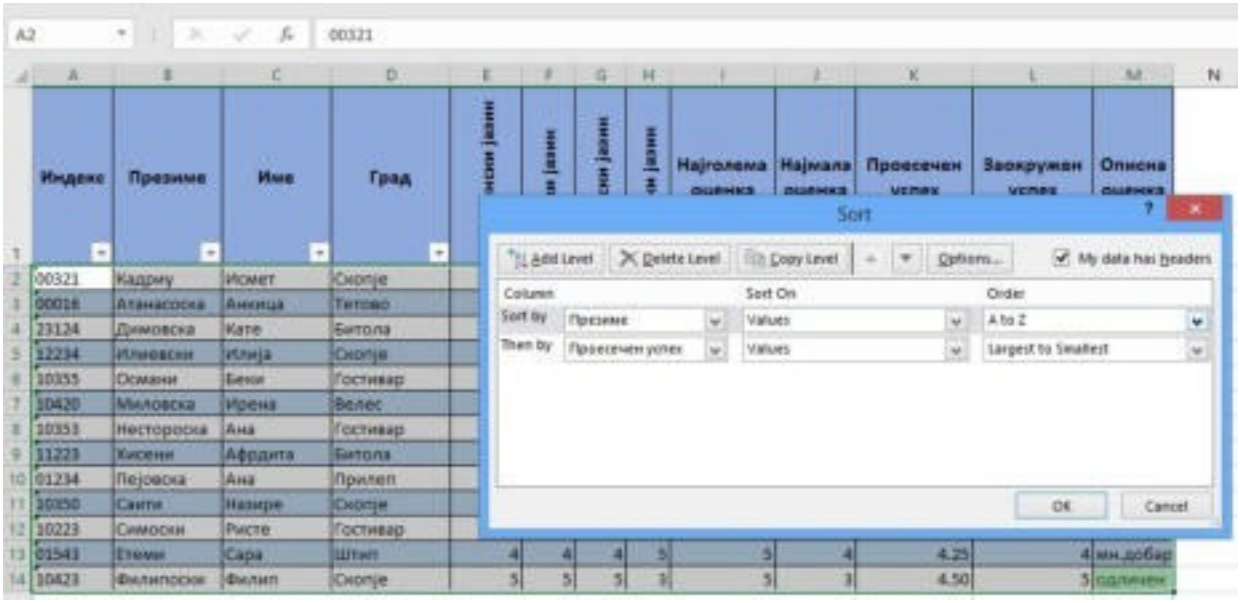


Custom Sort (Напредно подредување)

Ако сакаш да подредуваш **по повеќе правила**, кликни:

- **Sort → Custom Sort**
- Во прозорецот, можеш да:
  - Избереш **прва колона** (на пр. презиме)
  - Додадеш **втора колона** (на пр. успех)

- Одлучиш дали да е од најмало до најголемо или обратно

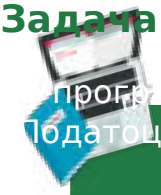


Обидете се! *(hint activity)*

Имаш табела со име и успех. Пробај да ги подредиш:

- Прво по **име на ученикот** (А до Ш)
- Потоа, **по успех** (5.0 до 1.0)

Задача



прогр
Податоци од библиотека“:

	A	B	C	D	E	F
1	Податоци од библиотека					
2	Ред бр	Лектира	Име на автор	Презиме на автор	Оддление	Изнајмена
3	1	Мајка	Јован	Јовановиќ - Змај	II	84
4	2	Зоки Поки	Оливера	Николовска	II	68
5	3	Поезија	Рифат	Кукај	III	35
6	4	Шеќерна приказна	Славко	Јанески	III	44
7	5	Сказна за детето Вилен	Глигор	Прличев	IV	54
8	6	Училиште	Драган	Лукиќ	III	30
9	7	Орхан	Нејати	Зекирија	IV	57
10	8	Летни цвеќиња	Наим	Фрашери	VI	67
11	9	Силјан Штркот	Марко	Цепенков	VII	46
12	10	Принцезата Арѓиро	Исмаил	Кадаре	VII	44
13	11	Бегалка	Видое	Подгорец	VIII	70

Да се изврши сортирање по категорија **„Лектира“** по азбучен редослед, а потоа по колку пати е **„Изнајмена“** започнувајќи од најголемата вредност кон најмалата!

ри го знаењето

1. Што е сортирање?
2. Која е постапката за сортирање на податоците по азбучен редослед?
3. Како можеме да извршиме сортирање на податоците по даден услов?

1.2 Креирање графикон

Прикажи ја табелата како СЛИКА: ГРАФИКОН!

 Замисли дека имаш табела со податоци, но сакаш брзо да разбереш кој ученик има највисок успех? Или колку ученици имаат 5ка?

Excel има **моќна алатка** што ги претвора податоците во **слика** – тоа се **графиконите**.

Што е графикон?

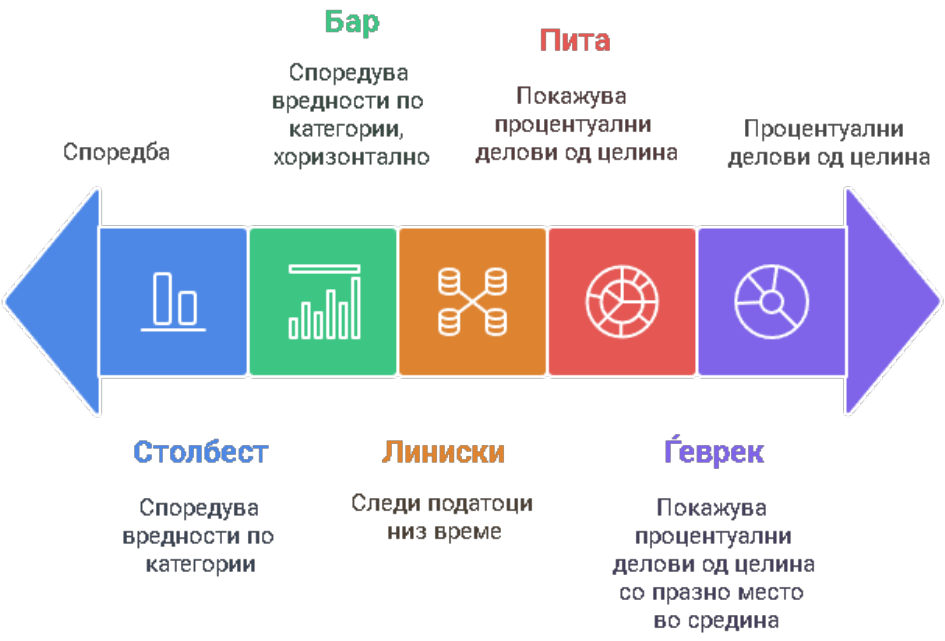
✏ Графикон е сликовит приказ на табеларни податоци.

 Наместо да гледаме бројки, гледаме *столбови, линии, парчиња пита...* – и така многу полесно забележуваме разлики и сличности.

📈 Видови графикони:

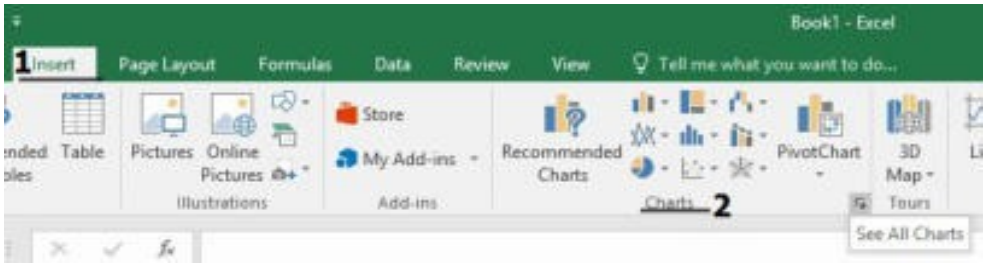
Тип на графикон	Кога се користи?
Столбест (Column)	Кога споредуваме вредности по категории (на пр. успех по ученик)
Линиски (Line)	Кога следиме податоци низ време (на пр. оценки по месеци)
Пита (Pie)	Кога гледаме процентуални делови од целина (на пр. процент на ученици со 5ка)
Ѓеврек (Doughnut)	Сличен на пита, но со празно место во средина
Бар (Bar)	Хоризонтални столбови – кога насловите се долги

Визуелизација на податоци: од споредба до процентуални делови од целина

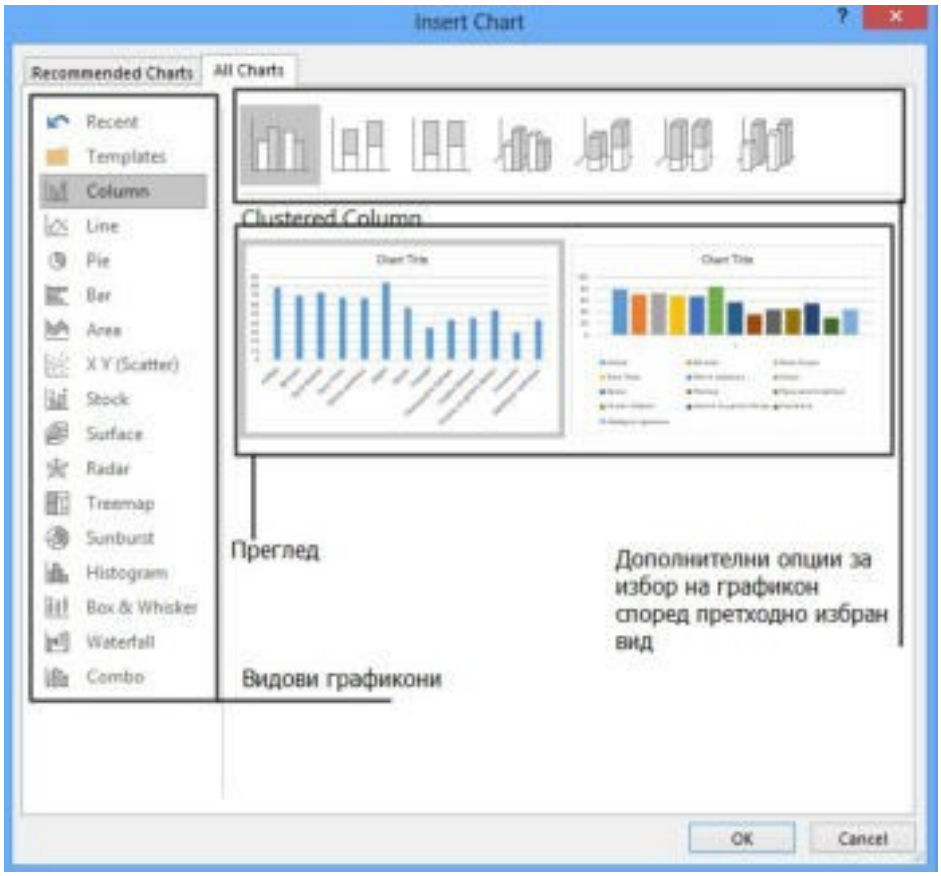


Обидете се! ***(Hint чекори)***

1. ✔ Селектирај ја табелата со податоците што сакаш да ги прикажеш;
2. 🖱 Отвори **Insert → Charts**;
3. 🖱 Одбери тип на графикон (на пример: *Column Chart*);
4. 🖱 Можеш да го уредиш графиконот: додади наслов, боја, легенда...



5. Од сликата, забележуваме дека при вметнување на графикон веднаш можеме да ја избереме категоријата, односно видот на графиконот или, пак, преку **Quick Launcher**, кој се наоѓа на левата страна на рубриката **Chart**, при што се прикажува следниот прозорец:



Со кликнување на копчето **OK**, графиконот се појавува во работниот лист во кој се наоѓа и изворот на податоци, односно изворната табела. Тогаш одлучуваме дали графиконот е прегледен да остане во истиот работен лист или, пак, да го пренесме во нов работен лист. Исто така, додека е селектиран графиконот се појавуваат менијата **Design** и **Format** кои овозможуваат форматирање и уредување на изгледот на графиконот. На пример, преку менито **Design** можеме да ги примениме следните можности:

- додавање елементи на графиконот;
- распоред на елементите на графиконот;
- избор на боја;
- стил на графиконот;
- избор на податоци;
- промена на видот на графиконот;
- префрлање на графиконот во нов работен лист/во

истиот работен лист со табелата.

Преку менито Format можеме да додадеме облици на деловите од

Графиконот, боја на линијата, обојување на елементите, стилови на буквите, димензии на елементите и сл.

Провери го знаењето

1. Што е графикон?
2. Опиши ја постапката за креирање графикон?
3. Кои видови графикони овозможува програмата Excel 2016?

4.Кои алатки може да се употребуваат од менито Design и Format за уредување и форматирање на графиконот?

🎮 Excel Master Challenges

🧩 Challenge 1: Тајната на Ќелијата A1

🎯 **Мисија**: Пронајди ја активната ќелија и промени ја содржината без да избришеш сè.

📋 **Задача**:

1. Отвори Vezba2.xlsx
2. Пронајди ја ќелијата со текст „Села“ и промени го името во „Селими“
3. Додај една буква во името „Јован“ → „Јованчо“
4. Зачувај го документот

🔁 *Повторува: активна ќелија, едитирање, статусна лента*

🧩 Challenge 2: Ќелиите Шифра

🎯 **Загатка**: Во работниот лист Sheet2, скриени се шифри во ќелии A3 до E3. Тие се низа на **парни броеви**. Мораш да ја откриеш шемата и да ја продолжиш низата!

✍️ **Инструкции**:

1. Во A3 внеси 2
2. Во B3 внеси 4
3. Со **Autofill**, пополни до E3

🔍 Потоа, пробај со имиња: Ана, Ана Марија, Марко (погледни дали Excel ја продолжува логиката)

🔁 *Повторува: Autofill, обрасци, типови податоци*

🧩 Challenge 3: Форматското фијаско!

🎯 **Задача**: Твојата табела изгледа ужасно. Форматирај ја така што ќе биде лесна за читање и убава.

📋 **Мисија**:

1. Подебели ги имињата на учениците
2. Центрирај ги сите податоци во табелата

3. Обои ја насловната редица во светло сина
4. Спои ги ќелиите A1:G1 и напиши „ТАБЕЛА НА УЧЕНИЦИ“

 Повторува: форматирање, порамнување, Merge Cells, боја на ќелија

 Challenge 4: Excel Escape Room

 Загатка: За да излезеш од табелата, мораш да ги сортираш податоците по успех, па по презиме.

 Чекори:

1. Во листот Успех, кликни на која било ќелија во колоната „Успех“
2. Сортирај по опаѓачки редослед
3. Додади второ сортирање: по презиме по азбучен ред

 Повторува: сортирање, Custom Sort

 Challenge 5: Графиконска детективска мисија

 Теоретска задача:
Најди која класа има најмногу ученици со просек 5. Користи графикон!

 Како:

1. Изработи табела: Колона A – Име, B – Просек
2. Внеси податоци за 6 ученици
3. Вметни **Column Chart**
4. Анализирај: кој е највисок столб?

 Повторува: визуелизација, Insert → Chart

 Boss Level Challenge: Табеларна тајна агенција

 Мисија: Сите податоци ти се расфрлани. Треба да:

1. Форматираш ја табелата како професионален извештај
2. Ги вметнеш формулите за: Количина x Цена = Вкупно
3. Изнесеш **просек** од вкупната сума (Hint: користи =AVERAGE())

 Ако го завршиш ова без грешка, стануваш:
 Excel Agent – Level 1

Дополнителни тест активности / мини-игри

Excel Bingo!

 Играчите добиваат табела 5x5 со Excel-команди/појмови (пример: Formula Bar, Ctrl, Fill Color, Cell, Autofill…). Наставникот кажува појми или ги опишува – учениците ги бојат на табелата. Прв со ред или колона = победник.

 Повторува: речник, препознавање алатки

Формула детективи

 Цел: Дадена е табела со пресметки – некои формули се скриени.

Задача:

- Во D2 пишува 300
- Во D3 пишува 450
- Во D4 треба да се внесе формула што ги собира D2 и D3

➡ Учениците да ја напишат точната формула =D2+D3

 Надградба: користи функции =SUM(D2:D4), =AVERAGE()

□ Лажен табеларен извештај □□

Цел: Пронајди 5 грешки во табелата:

- Погрешен формат (пример: датум напишан како текст)
- Несортирани податоци
- Формула без =
- Неправилно споени ќелии
- Несреден фонт

➡ Учениците треба да ги најдат и поправат грешките.

□ *Повторува*: форматирање, формули, уредување

□ Excel Скршен Телефон □

Како функционира:

1. Првиот ученик внесува податоци и користи Autofill
2. Вториот применува форматирање
3. Третиот додава формули
4. Четвртиот прави графикон

➡ Секој има 2 минути и мора да го продолжи од претходниот чекор.

□ *Цел*: Тимска работа + повторување целиот циклус

□ Четка на Excel – Дизајнирај визит-картичка! □

Задача:

- Секој ученик во табела да дизајнира “визит-картичка” со име, адреса, телефон, итн.

- Да користи Merge Cells, боја, фонт, рамки

 Повторува: уредување, спојување ќелии, форматирање

Excel Мистерија – Кој го скрил податокот?

Задача:

- Скриј колона/ред со важен податок
- Постави филтри и сортирање
- Учениците да ја откријат скриената ќелија (пример: вредност = 0, или име на скриен ученик)

 Повторува: скривање/откривање, филтри, сортирање

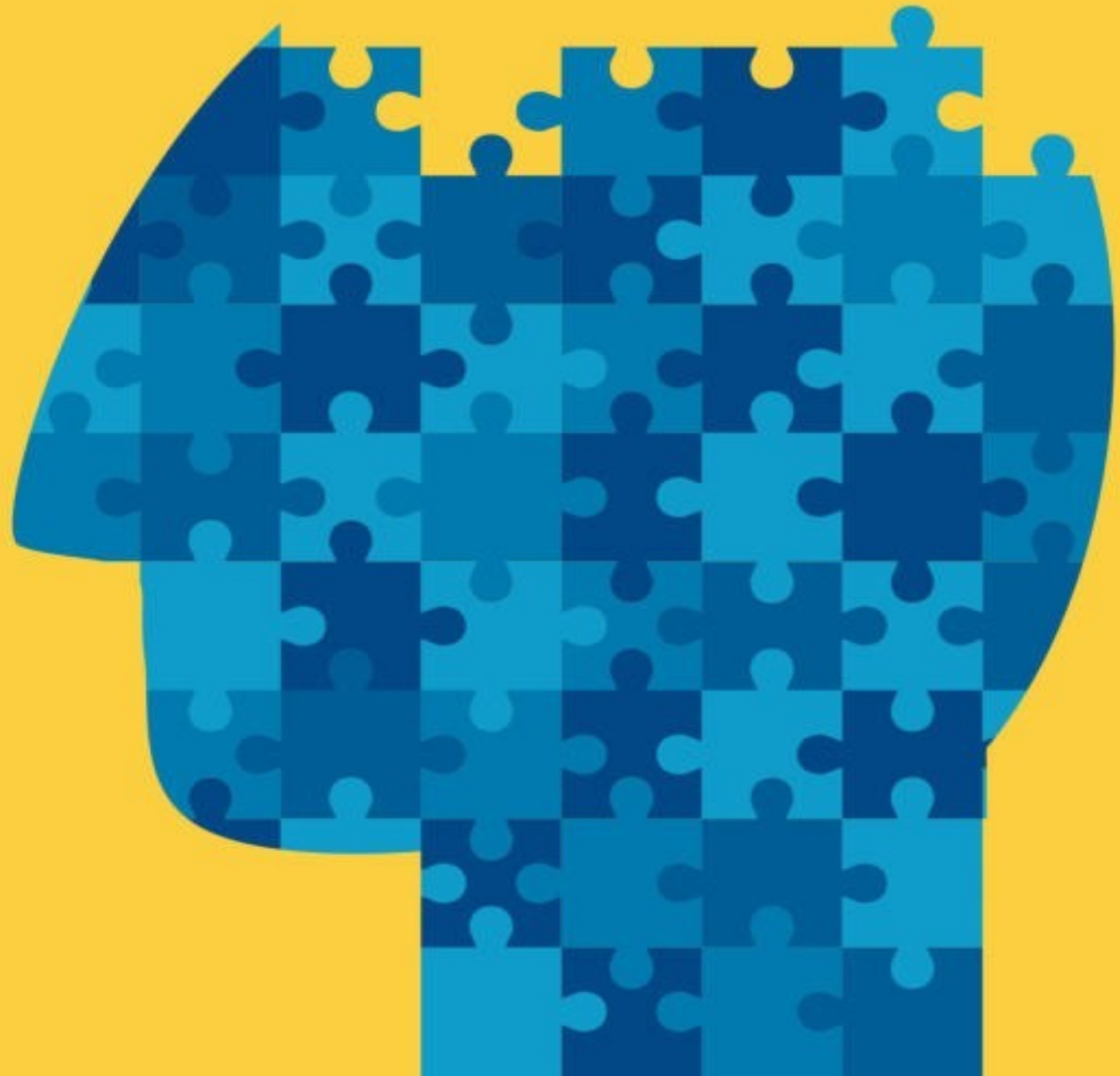
Тест – 10 брзи Excel прашања (мешани типови)

1. Што е Autofill?
2. Што значи “A1:A5”?
3. Што прави формулата `=A2*B2`?
4. Како се менува бојата на текст во ќелија?
5. Кој симбол се користи за започнување формула?
6. Дали можеш да внесеш и текст и број во иста ќелија?
7. Што прави "Merge Cells"?
8. Што е "Wrap Text"?
9. Како се сортира табела по азбучен ред?
10. Дали можеш да имаш повеќе работни листови во еден Excel документ?

Запознавање со информатички концепти преку решавање логички натпреварувачки задачи

Запознавање со информатички концепти преку решавањена логички натпреварувачки задачи
Анализа и решавање на логички натпреварувачки задачи
Поврзаност на задачата со концепти од компјутерската наука – информатички концепти
- Структури на податоци
- Бинарни броеви
- Криптографија

ДА ПОВТОРИМЕ! ДА УВЕЖБАМЕ!



🧠 Запознавање со Информатика преку Логички Загатки

Дали знаеш дека информатиката не е само за компјутери? Таа ни помага да размислуваме **попрецизно, попаметно и покреативно**!

Во оваа лекција ќе учиме:

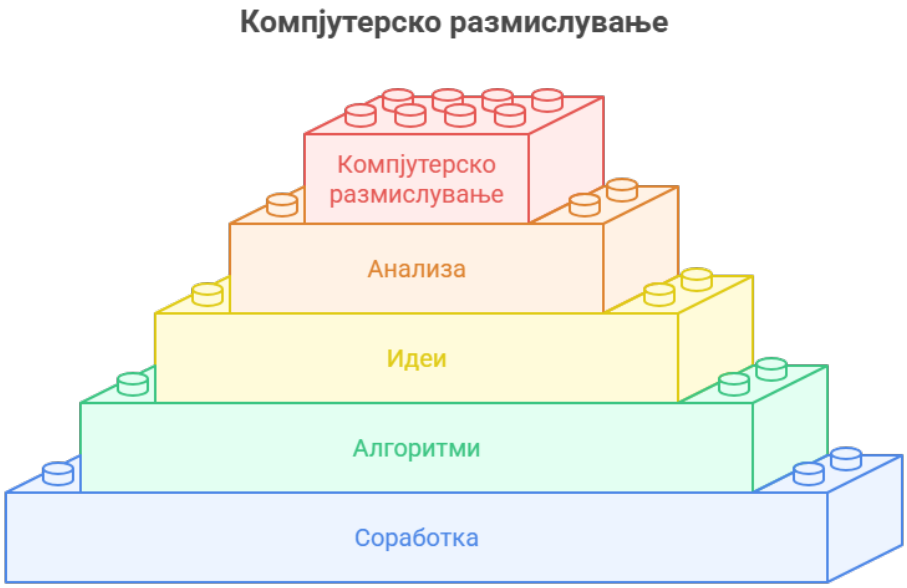
- Како да **креираме табели и низи** со различни податоци 📊
- Како да користиме **формули и функции** за пресметки 🧮
- Како да ги **украсиме табелите** со бои, фонтови и графикони 🎨
- И најважно – како да **размислуваме логично и паметно!**

Ова се нарекува **компјутерско размислување**. Тоа не значи само да работиме со компјутери – туку да **ги решаваме проблемите** на паметен начин!

🤔 Како да размислуваме како компјутер?

- 🧩 Да бидеме **логични** – да има смисла тоа што го правиме
- 🔁 Да користиме **алгоритми** – чекор по чекор решенија
- 💡 Да размислуваме **иновативно** – нови идеи!
- 🤝 Да работиме **во тим** – заедно сме посилни
- 🎯 Да бидеме **прецизни и креативни** во исто време

Логичкото размислување е како **ментална вежба** – што повеќе го користиме, толку подобро ни оди!



🧠 Обиди се да размислуваш... логички!

Знаеш што е логика?
Тоа е кога нешто **има смисла, врска и точен редослед** – како мали парчиња од сложувалка што се вклопуваат!

🧩 **Логичкото размислување** ни помага да решаваме задачи, проблеми и секојдневни ситуации со:

- факти 📚
- докази 🔍
- анализа, а не со емоции 🙅‍♂️

🤔 А што е разликата меѓу логичко и критичко размислување?

🧠 *Логичко размислување* = решаваме проблем чекор по чекор со точни информации.
🔍 *Критичко размислување* = анализираме, споредуваме, оценуваме различни идеи и мислења.

И двете се важни, но логиката не прифаќа „мислам дека“ — бара „знам затоа што“! 💡

🧩 Пример: СУДОКУ!

Во судоку имаш само неколку броеви – а мораш да ги откриеш сите! Ако згрешиш еден, целата табела се расипува. Тоа е логичка задача!

🧠 Логика во реалниот живот:

- пресметуваш дали имаш доволно пари во маркет
- решаваш како да си ги организираш обврските
- проверуваш дали некоја вест е вистинита или лажна

⚠ Внимавај:

- Не заклучувај само по свој вкус (пример: мирисот на ручекот не значи дека е нездрав).
- Не претпоставувај – провери!
- Постави прашања.
- Направи стратегија.
- Собери информации, па дури потоа носи одлука.

Зборовите се важни!

Кога решаваш задача или проблем, **секоја мала разлика во зборовите може да значи нешто големо**. Да разјасниме:

- „**Неопходно**“ значи дека нешто **мора** да се направи.
- „**Доволно**“ значи дека може и со **минимален труд** да стигнеш до целта.

🔍 Ако не ги разбереме точно зборовите – може да **направиме грешка во размислувањето**. Затоа, логичното размислување бара да бидеме прецизни!

🚀 Логика пред програмирање!

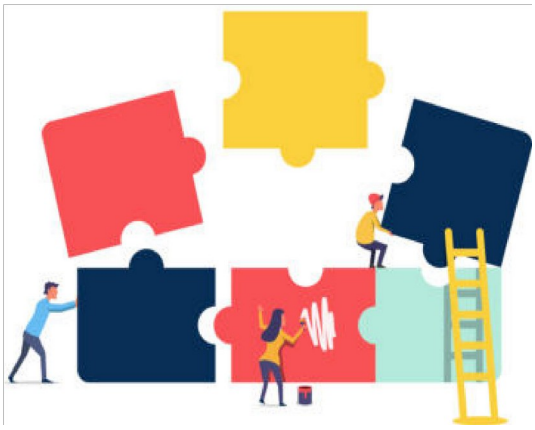
Пред да научиме да програмираме, мора да научиме да **размислуваме јасно и логично**. Програмирањето е само алатка – **логиката е нашиот мозочен код!**

Еве неколку *кул зборчиња* кои често ќе ги среќаваш во информатика:

- **Компјутерско размислување** – паметен начин на решавање на проблеми
- **Анализа** – гледаме детали, бараме врски
- **Програмирање** – пишување код за да направиме нешто
- **Бинарни броеви** – јазикот на компјутерите: 1 и 0
- **Кодирање и криптографија** – скривање пораки, како тајни агенти
- **Слободен софтвер** – програми кои можеш да ги користиш и споделуваш слободно

🧠 Запомни:

- Логиката не користи „мислам“ – туку „знам затоа што проверив“.
- Не се води по чувства, туку по **докази и факти**.
- Со логичко размислување – го чистиме патот до точно решение!



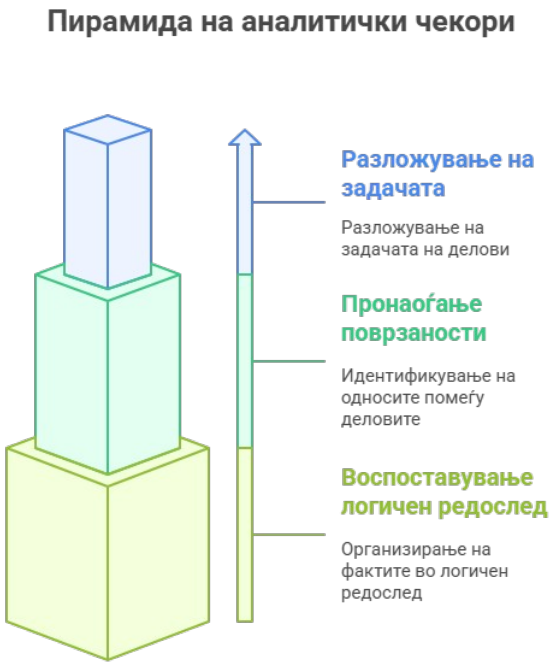
🧩 Анализа – првиот чекор кон решението!

Кога имаме некоја логичка задача, не трчаме веднаш да ја решиме. Прво треба да направиме **анализа** – тоа значи внимателно да ја разгледаме целата задача, дел по дел.
🎁 *Како да ја „распакуваме“ – како подарок во повеќе слоеви!*

🔍 Трите чекори на анализа:

1. **Разложи ја задачата на делови** – Што точно знаеме?
2. **Пронајди поврзаности** – Што се поврзува со што?

3. **Постави го логичниот редослед** – Кој факт води до кој заклучок?


Пирамида на аналитички чекори
Разложување на задачата
Разложување на задачата на делови
Пронаоѓање поврзаности
Идентификување на односите помеѓу деловите
Воспоставување логичен редослед
Организирање на фактите во логичен редослед

 Ако следиме вакви чекори, многу полесно ќе стигнеме до точното решение!

Обиди се!

 Што е спротивно од анализа?
Анализата значи „расклопување“ – а спротивното е **синтеза** – „спојување на делови во целина“.

Пример задача:

Астрет, Јован и Сами – сите се добри атлетичари.

- Знаеме дека **Астрет е најбрз** – супер! Тој оди во квадратчето „најбрз“.

- За **Јован** пишува дека **не е најспор** – значи не оди таму.
  Сега останува само едно место празно. Според логиката – тоа мора да е за **Сами**, и тој е најспор.

Решението е… Сами!

Ајде да ги повториме фактите:

- ♂ Јован **не е најспор**
- ♂ Астрет **е најбрз**
- ♂ Сами **не е најбрз**

 Со ова:

- „Најбрз“ го доби Астрет
- „Јован не е најспор“, па не може да оди таму
- Останува „Сами“ да биде **најспор** – и така го решаваме проблемот!

 Одговор: Сами е најспор.

Користевме техника: апстракција

Што значи тоа?
 Апстракција е кога ги вадиме **само важните работи** од задачата – а ги **игнорираме непотребните**.

 Истото го прават и програмите!
Кога решаваме проблем во информатика, **се фокусираме на корисните податоци**, а сите непотребни информации ги оставаме на страна.

Логика на дело: Играј СУДОКУ со фигури!

Знаеш ли дека Судоку не мора да биде само со броеви? Може да се игра и со **геометриски фигури**!

 Имаме табела каде некои полиња веќе содржат:

- **Круг**
- □ **Квадрат**
- **Триаголник**
- **Ѕвезда**

Но... има и **празни полиња**. Нашата задача е:
 Да ги пополниме така што **во секоја редица и колона ќе има само по една фигура од секој вид!**

Да најдеме каде оди кругот!

- **Круг** веќе има во:
  - Прва редица
  - Трета редица
  - Четврта редица

 Значи останува само **втората редица**!

Сега проверуваме **во која колона**:

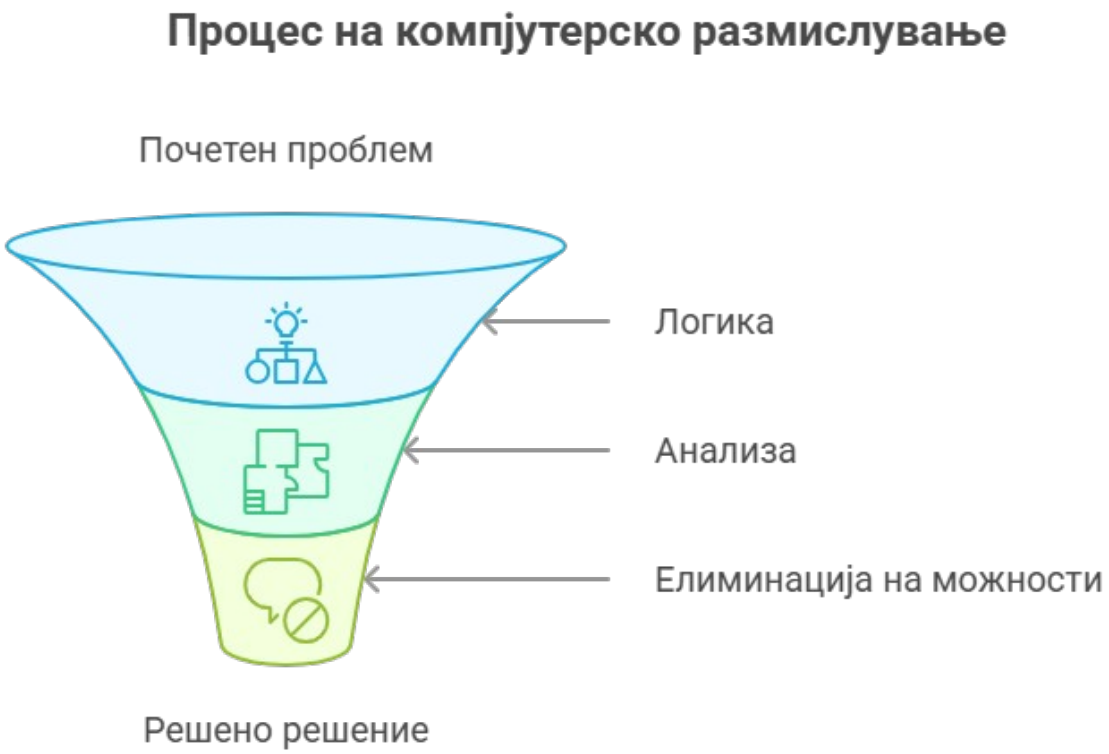
- Прва, втора и трета колона **веќе имаат круг**
- Значи останува само **четвртата колона**!

 Точниот одговор е:
 Кругот оди во **втора редица, четврта колона**.

 Ова го решивме со:

- Логика
- Анализа
- Елиминација на можности

И тоа е суштината на **компјутерското размислување**!



🧠 2.2 Како се поврзуваат логичките задачи со информатика?

Кога решаваме логички задачи, ние учиме како да размислуваме **јасно**, **точно** и **паметно** – баш како што размислуваат **компјутерите!**

🧠 Затоа велиме:
🧠 **Логичкото размислување** е **првиот чекор кон програмирањето**!

🧠 Што е програмирање?

Програмирање е процес во кој се пишуваат **инструкции** (кодови) кои компјутерот ги следи за да направи нешто – на пример:

- Игра да работи 🎮
- Алатка да пресмета нешто 🧮
- Апликација да испрати порака 📩

 Луѓето што го прават ова се нарекуваат: **програмери**.

2.2.1 Што се структури на податоци?

Замисли дека сакаш да подредиш предмети по ред:

- Книга, молив, гума, лењир...

 Структура на податоци е начин на **средување и чување** такви информации.

Најпознати се:

- **Низа (array)** – ред од податоци: [молив, гума, лењир]
- **Листа (list)** – како список на задачи
- **Стек (stack)** – последното ставање е прво вадење
- **Ред (queue)** – прв влегува – прв излегува
- **Бинарно дрво (binary tree)** – како семејно стебло!

Примери од секојдневниот живот:

Низа од инструкции:

- Прво вклучи го компјутерот
- Потоа отвори Word
- Напиши наслов
- Зачувај ја датотеката

 Упатство до некоја локација:
[Од дома] → [сврти лево] → [оди право] → [сврти десно] → [училиште]

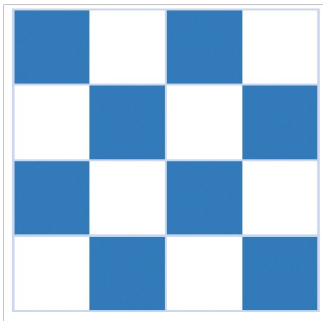
 Пополнување на формулар:
Име → Презиме → Адреса → Телефон

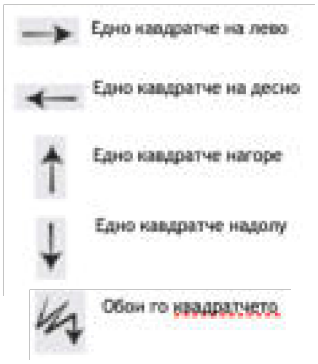
Задача: Креирање низа од чекори – Поместување на дискови

Замисли дека имаш три диска со бројки 1, 2 и 3, и треба да ги префрлиш од едно место на друго. Важно е да ги поместуваш **во точен редослед**!

Чекори:
1 Дискот 1 го ставаме во третата колона.
2 Дискот 2 го ставаме во втората колона.
3 Потоа, дискот 1 го префрламе над дискот 2 во втората колона.
4 Дискот 3 го ставаме во третата колона.
5 Сега, дискот 1 го враќаме во првата колона.
6 И на крај, дискот 2 го ставаме над дискот 3 во третата колона.

 Ова е пример за **низа од чекори** — серија од јасни инструкции кои помагаат да се стигне до целта без грешки!





Задача: Кулите на Ханои – како да ги преместиш трите дискови?

Сте слушнале ли за играта **Кулите на Ханои**? Еве еден пример со три дискови!

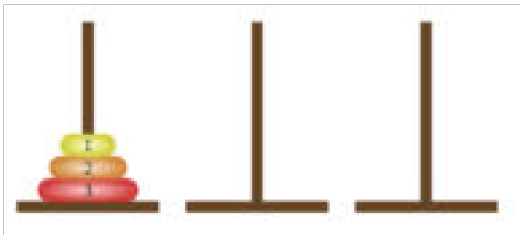
- Во чекор 7, го префрламе дискот број 1 во третата колона, **над дискот број 2**.
- Со само **седум чекори**, ги префрливме сите дискови од првиот до третиот ред, без да го смениме нивниот редослед.

Оваа игра ни помага да сфатиме важна структура од информатиката наречена **стек (куп)**.

Што е стек?

Стек е начин на чување на информации каде што важи правилото: **"Прв влегува, последен излегува"**. Значи, секогаш го земаме последниот додаден диск, како што е при преместувањето во играта.

Правилата на Кулите на Ханои ги одразуваат токму овие карактеристики, па затоа стекот е совршена структура за решавање на вакви задачи!



Обидете се!

Заедно со наставникот по биологија, обидете се да направите **низа** која го прикажува животниот циклус на пеперутката.

- Колку **периоди** има животниот циклус на пеперутката?
- Дали може да се прескокне некој од овие периоди?

Животниот циклус на пеперутката има неколку важни фази кои секогаш се случуваат по ред: јајце, ларва, лулка и пеперутка. Секоја фаза е важна и не може да се прескокне!

Обидете се да ги наредите фазите во точниот редослед и направете низа со тие чекор

2.2.2 Бинарни броеви

Да се потсетиме!
Кој е јазикот на компјутерите?

Податоците во компјутерите се претставуваат со **бинарен бројен систем**, кој се состои од две цифри:

- **0** и **1**.

Една цифра 0 или 1 се нарекува **бит**.

Со еден бит можеме да претставиме двојни вредности, како што се:

- „точно“ или „неточно“
- „да“ или „не“
- „зафатено“ или „слободно“

Податоците во компјутерите се запишуваат како низи од 0 и 1, што се нарекуваат **бинарни броеви**.

Претворањето на букви или зборови во бинарен код значи дека секоја буква се претставува со свој бинарен запис.

Секој природен број може да се претстави како бинарен број, користејќи ја основата 2 и степените на 2 (0, 1, 2, 3 и така натаму).

На пример, бројот 5 во бинарен запис е:
0101

(Значењето: $0\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0 = 4 + 0 + 1 = 5$)

Натпреварувачка задача:

Учениците од VII одделение решавале тест по географија кој содржел прашања со повеќе понудени одговори. Наставничката точните одговори ги бележела со **1**, а неточните со **0**, во табелата прикажана подолу.

Целта е да се креира извештај за тоа:

- Кој дел од материјалот е најслаб усвоен, а кој најдобро,
- За да се предложат дополнителни начини за учење.

Одговорете на следните прашања, користејќи ги податоците од табелата:

1. Кое прашање го одговориле најмногу ученици?
2. Кое прашање го одговориле најмалку ученици?
3. Кој ученик има највисок резултат?
4. Кој ученик е најслаб?
5. Кои ученици треба да посетат дополнителна настава според резултатите?

2.2.3 Криптографија

Криптографијата е научна дисциплина која се занимава со проучување на методите за испраќање и примање пораки на начин кој е разбирлив само за оние за кои е наменета.

Цели на криптографијата:

- Заштита на податоците од неовластен пристап
- Сигурна и заштитена комуникација
- Веродостојност и тајност на пораките
- Проверка на потеклото на информациите

Цели на криптографијата
Заштита на податоците
Заштита на податоците од неовластен пристап.
Интегритет на пораките
Веродостојност и тајност на пораките.
Сигурна комуникација
Сигурни и заштитени методи на комуникација.
Проверка на потеклото
Проверка на потеклото на информациите.

Криптографијата се користи при:

- Работа со кредитни картички
- Компјутерски лозинки
- Електронска трговија
- И други активности каде што е важна заштитата на корисникот

Криптографски процес

Криптографијата вклучува две основни трансформации на податоците или пораките:

- **Шифрирање (енкрипција)** — процес на трансформација на пораката во шифра (неразбирлива форма)
- **Дешифрирање (декрипција)** — обратен процес кој ја враќа шифрираната порака во оригинална, разбирлива форма

Овој процес е прикажан на следната слика (обично дијаграм со пораката, шифрирање, пренос и дешифрирање).

Примери за криптографија

- **Морзеова азбука** — користена за телеграфска комуникација, каде секоја буква, број и интерпункциски знак се претставени со комбинации од точки и црти.
- **Цезарова шифра** — една од најпознатите и најчесто користени техники за шифрирање, базирана на поместување на буквите во азбуката за фиксна вредност.

Практична задача:

Ајде да дешифрираме порака користејќи го Цезаровиот модел на шифрирање!

Ова направете го на час!

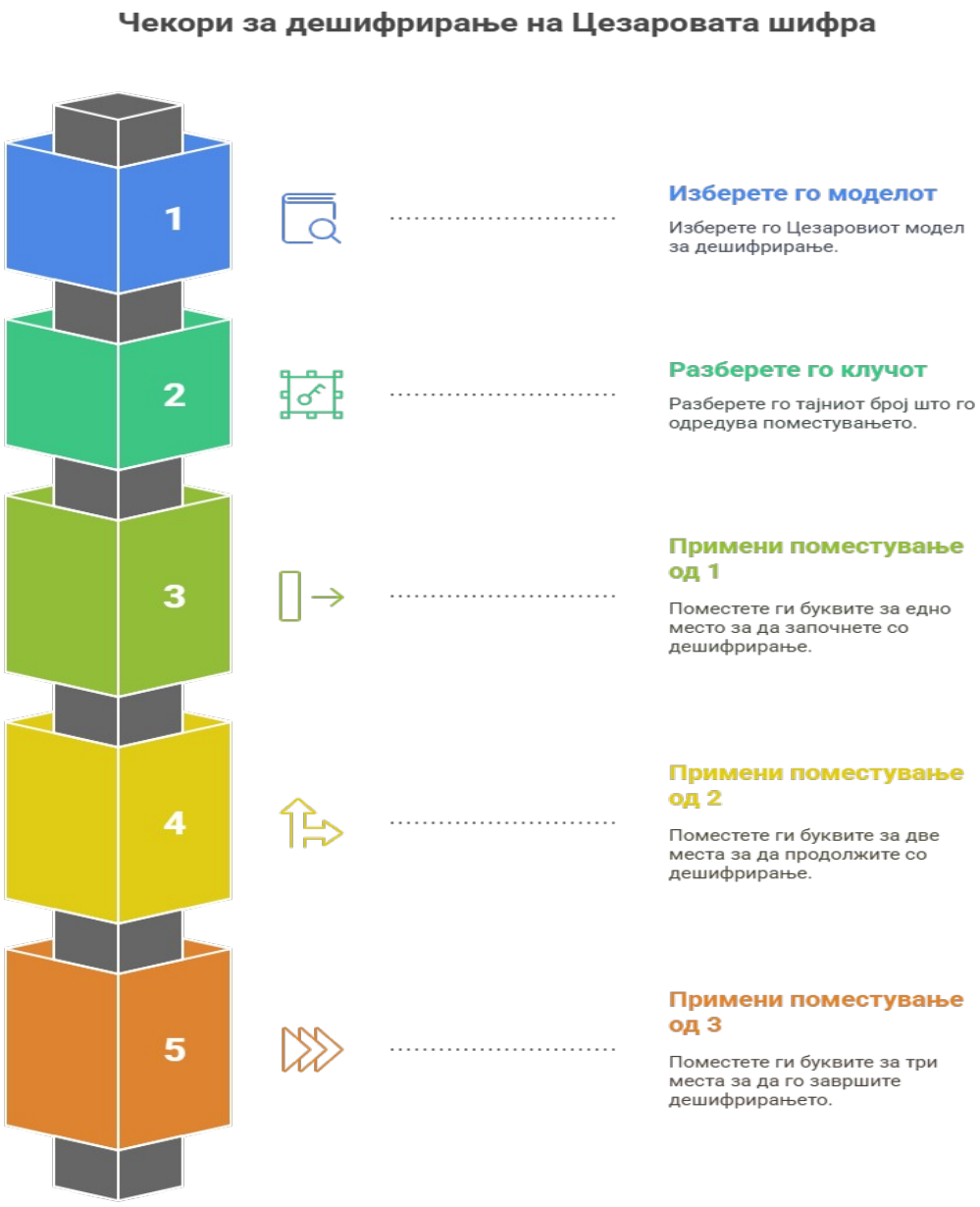
Задача: Шифрирана лозинка — Цезаров модел

Јана и Азра се другарки од VII одделение и треба да креираат проектна задача на тема **„Безбедност при користење на интернет“**. За таа цел им е потребен интернет пристап.

Но, при форматирање на компјутерот, лозинката за пристап до интернет е избришана. Братот на Јана ја памети лозинката, но сака да си поигра со нив и им дал **шифрирана лозинка** заедно со неколку инструкции за дешифрирање.

Упатства за дешифрирање:

- Дешифрирањето ќе го извршиме со **Цезаров модел** (Цезарова шифра).
- Цезаровата шифра користи таен број (клуч) кој ја одредува вредноста на поместувањето на буквите.
- Во оваа задача, користени се неколку цели броеви како клучеви за поместување: **1, 2 и 3 места**.


Чекори за дешифрирање на Цезаровата шифра
1
2
3
4
5
Изберете го моделот
Изберете го Цезаровиот модел за дешифрирање.
Разберете го клучот
Разберете го тајниот број што го одредува поместувањето.
Примени поместување од 1
Поместете ги буквите за едно место за да започнете со дешифрирање.
Примени поместување од 2
Поместете ги буквите за две места за да продолжите со дешифрирање.
Примени поместување од 3
Поместете ги буквите за три места за да го завршите дешифрирањето.

Како да започнеме?

1. Во табелата ја запишуваме азбуката (пример: А, Б, В, Г, ...).

2. За секоја буква од шифрата применуваме поместување назад (дешифрирање) за бројот места што е даден како клуч (1, 2 или 3).
3. На овој начин ја враќаме шифрираната порака во оригиналниот текст — лозинката.

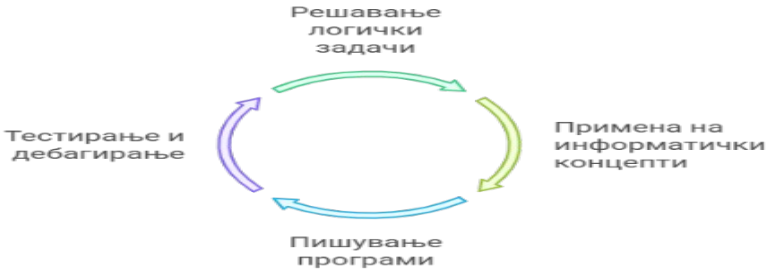
Практична задача:

- Добивме шифрирана порака и знаеме кои се клучевите за дешифрирање (1, 2 и 3).
- Помогнете си со логичко размислување за да ја најдете вистинската лозинка.

Запомнете:

- Логичкото размислување и решавањето логички задачи се чекор пред програмирањето.
- Програмирањето е процес на пишување на програма, а програмерите се луѓе кои ги креираат тие програми.
- Логичките задачи и нивното решавање се тесно поврзани со информатички концепти како:
    - низи
    - листи
    - стекови/купови
    - редови
    - приоритетни редови
    - бинарни броеви
    - криптографија и други.

Циклус на логичко размислување и програмирање



ДА ПОВТОРИМЕ! ДА УВЕЖБАМЕ!

Задача 1

Креирајте низа од инструкции кои ќе го прикажат патот од влезот на училиштето до вашата училница.

Задача 2

Природните броеви **12, 18 и 36** претворете ги во бинарни броеви!

Задача 3

Бинарниот број **11100111** претворете го во природен број!

Задача 4

Бројот **236** да се претвори во бинарен број!

Задача 5

Креирајте шифрирана порака употребувајќи го **Цезаровиот модел**!

Задача 6

Истражете ги карактеристиките на игрите **Pacman** и **Tetris**.
Со кои информатички концепти можеме да ги поврземе? Објаснете!

Задача 7

Алма сака да ѝ испрати порака на Ана: **„ИЗЛЕГУВАМЕ“** со Цезаров код. Како гласи пораката која Алма ќе ја испрати?

Задача 8

Стефан добил порака од Амир: **„ПГОП“** кодирана со Цезаров код. Што сакал да му каже Амир на Стефан, ако клучот е број три (3)?

Задача 9

Јасмин испратил порака на Томи:
„УНК ОТДSА“.

Но, пораката ја пресретнал Елвин. Елвин знае дека Јасмин и Томи се допишуваат на начин што секоја буква од азбуката се поместува за шест (6) места.

Прашање:
Која порака ја добил Елвин при дешифрирањето?

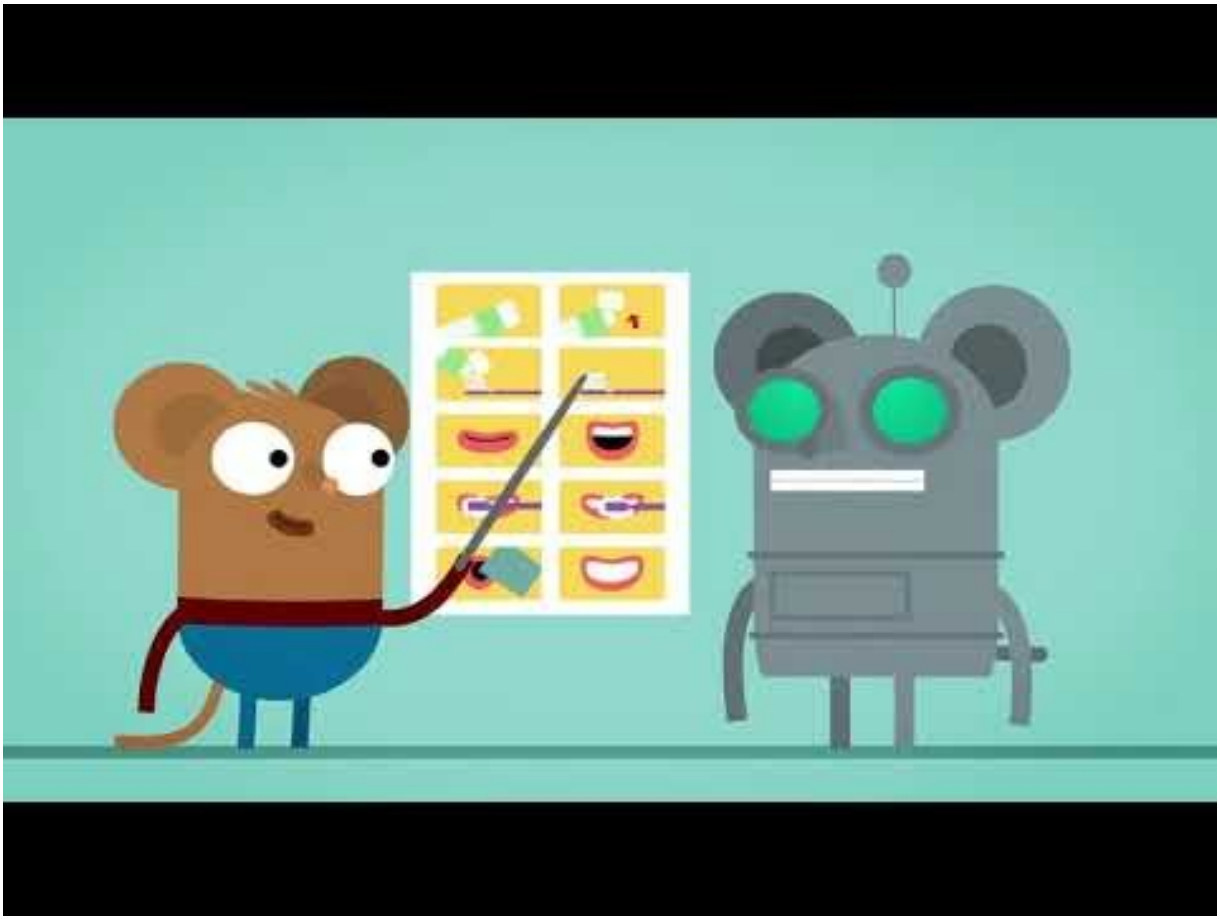
Дополнителни задачи

За увежбување на логичкото размислување и решавањето на логички натпреварувачки задачи, може да ги користите следните прирачници и онлајн ресурси:

- Прирачници со задачи и објаснување од Дабар: www.talent.mk
- Интернет страници за решавање логички натпреварувачки задачи и игри:
    - www.bebras.org
    - BBC Bitesize - Logical Thinking

- http://digit.mile.mk/

Погледнете го ова на часот :



Scratch

Напредно програмирање во визуелна околина

Вовед

Scratch е визуелна програма за учење програмирање. Наместо да пишуваме код, **спојуваме блокчиња** што изгледаат како сложувалка.

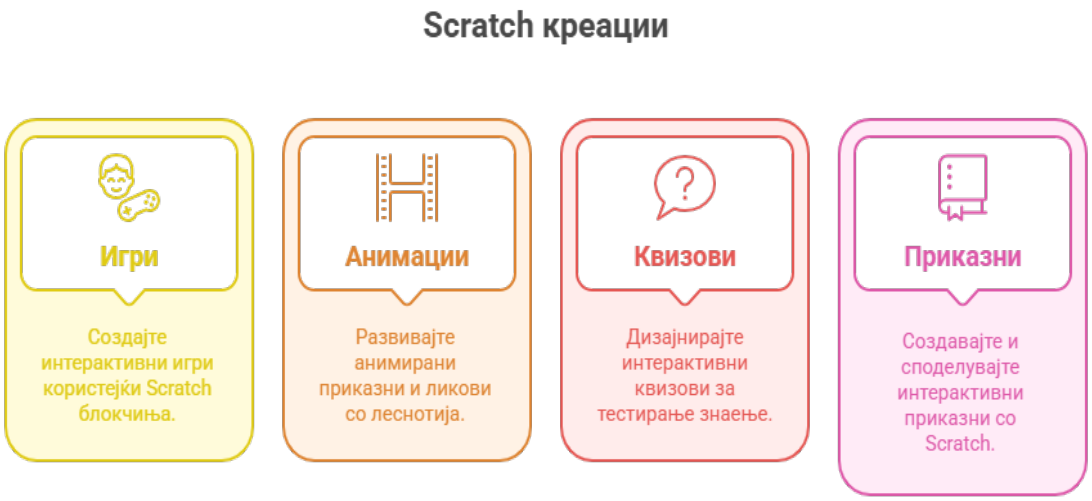
Со Scratch можеме да создадеме:

- игри
- анимации
- квизови
- приказни

Сè што ќе замислиме – можеме да го претвориме во програма!

Што учиме во оваа тема?

- Како се користи Scratch
- Како работат **блоковите со наредби**
- Како да направиме **интерактивна програма**
- Како да решаваме **посложени задачи со логика и настани**



🧠 3. Вовед во програмирање во визуелна околина

🤔 Што е програмирање?

Програмирање е **вештина на логичко размислување** и решавање проблеми со пишување **инструкции** што компјутерот ги извршува.

Целта: да се создаде точен сет на чекори (инструкции) кои водат до посакуван резултат – игра, анимација, апликација или некоја друга функција.

📌 Програма е:

- Сет на наредби што компјутерот ги следи.
- Решение за даден проблем или задача.

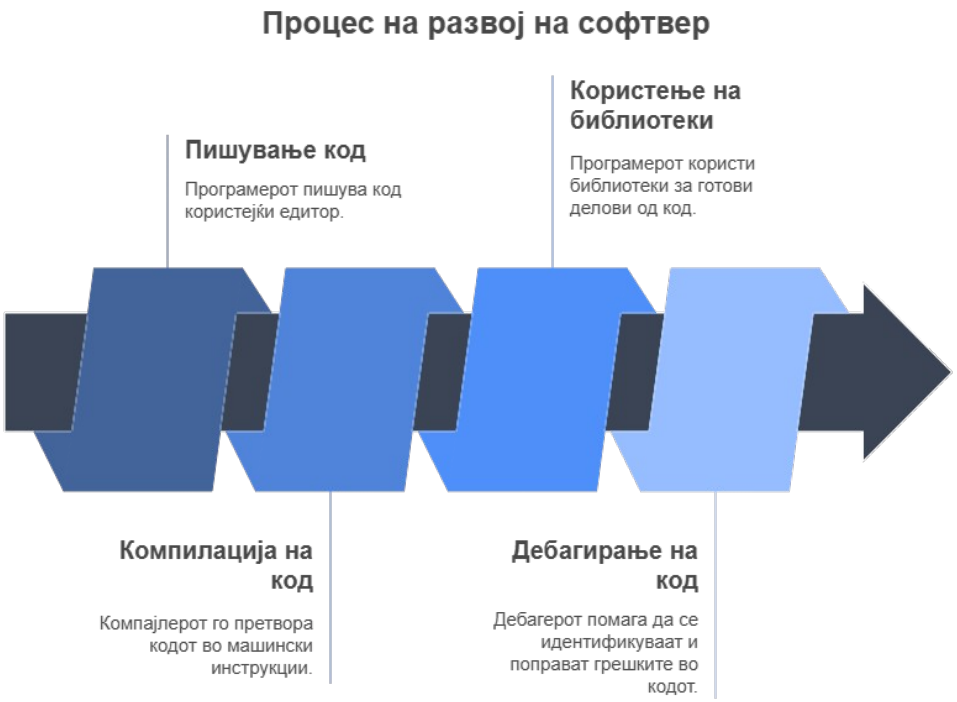
🧠 Програмирањето и логиката

Логичкото размислување е **прв чекор кон програмирање**. Затоа, натпреварувачки задачи и логички игри се важни при учење.

Како се изработува програма?

Програмерот користи:

- ✏ **Едитор** – за пишување на кодот
- ⚙ **Компајлер** – го претвора кодот во машински инструкции
- 📚 **Библиотека** – со готови делови од код
- 🐞 **Дебагер** – за откривање и поправка на грешки


Процес на развој на софтвер
Пишување код
Програмерот пишува код користејќи едитор.
Користење на библиотеки
Програмерот користи библиотеки за готови делови од код.
Компилација на код
Компајлерот го претвора кодот во машински инструкции.
Дебагирање на код
Дебагерот помага да се идентификуваат и поправат грешките во кодот.

Зошто визуелна околина?

Во **визуелно програмирање**, како Scratch:

- се користат **блокови** (слични на сложувалки)
- се програмира со **ликови, позадини, звуци и движења**
- програмирањето е **забавно и лесно за учење**

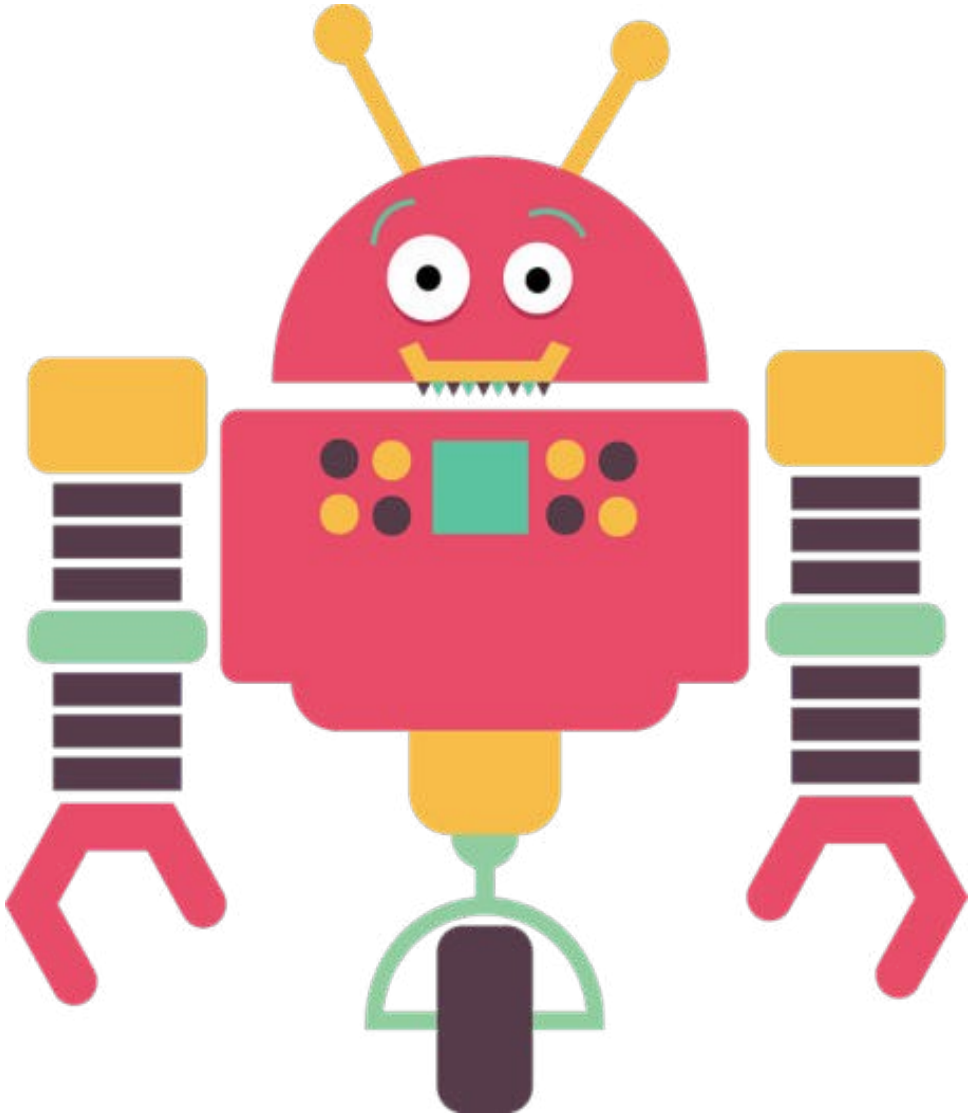
Scratch – визуелен јазик

Scratch е визуелен јазик во кој:

- се **креираат игри, приказни и анимации**
- се работи со **фигури (спрајтови)** и **настани**
- се развива **алгоритамско размислување**

Во оваа тема ќе научиме:

- Како изгледа работната околина на Scratch
- Како се изработуваат проекти
- Како се применува логика преку визуелни инструкц



3.1 Графичко програмирање

Запознавање со програмата Scratch

Што е графичко програмирање?

Графичкото (визуелно) програмирање е **забавен и лесен начин** за учење програмирање со користење на **слики, блокови и визуелни елементи**, наместо текстуални команди.

Карактеристики:

- Сите алатки и кодови се **графички прикажани**
- Се користат **фигури (ликови), звуци, позадини, движења**
- Учениците **развиваат креативност, логика и тимска работа**

Scratch е:

- **Визуелен програмски јазик**
- Се користи за: ❇ анимации ❇ игри ❇ приказни ❇ музика
- Се програмира со **влечење блокови** кои изгледаат како сложувалка

Каде може да го користиме Scratch?

- **Онлајн**: scratch.mit.edu
- **Офлајн**: Преземање бесплатна верзија од официјалната страница

3.1.1 Стартување на Scratch (офлајн)

Со двоклик на иконата на Scratch (на работната површина), се отвора **работниот прозорец**.

Работниот прозорец содржи:

- **Насловна лента** – го покажува името на програмата

- **Копчиња** – за затворање, минимизирање, максимизирање
- **Јазик** – избор на јазик на менијата (вклучува македонски)
- **File мени** – отворање и зачувување проекти
- **Мени лента** – пристап до алатки и наредби
- **Работна површина** – тука се поставуваат фигури (ликови)
- **Избор на фигури и позадини** – додавање елементи во проектот
- **Палета со наредби** – блокови за движење, звук, настани и сл. (боите покажуваат различни категории)
- **Простор за програма** – тука се **влечат и редат блоковите** кои ја создаваат активноста

Во Scratch не мора да пишуваме код – само ги влечеме блоковите и ги комбинираме!
На овој начин, учиме **логика, алгоритми и креативно решавање на задачи.**

3.1.2 Запознавање со основните алатки на Scratch

Објекти (Sprites)

Кога започнуваме со креирање програма во Scratch, првото што го гледаме е **ликот (sprite)** на екранот – тоа е **фигурата која ја програмираме** да извршува активности.

- **Првиот лик** е логото на Scratch.

- Нов лик се додава со клик на иконата
  Choose Sprite From Library.

Ликовите се поделени по категории:
животни, луѓе, фантастични, храна итн.

Начини за додавање објекти (sprites):

- Од **Scratch библиотеката**
- Од **компјутерот**
- Со **камера**
- Со **цртање сопствен лик**

Позадина (Background)

Позадината на сцената се менува **исто како и објектите**.

Додавање на позадина:

- Клик на иконата за **избор на позадина**
- Избор од **Scratch галерија**
- Од **компјутер**
- **Креирање сопствена тема**

Позадината му дава атмосфера и контекст на проектот.

Наредби (Искази)

Исказите се **инструкции што му кажуваат на компјутерот што да направи**.

Во Scratch, тие се **блокови од код** кои:

- Се **влечат со глувчето**
- Се **поврзуваат** како сложувалка

 Сите искази се наоѓаат во делот **Scripts**.

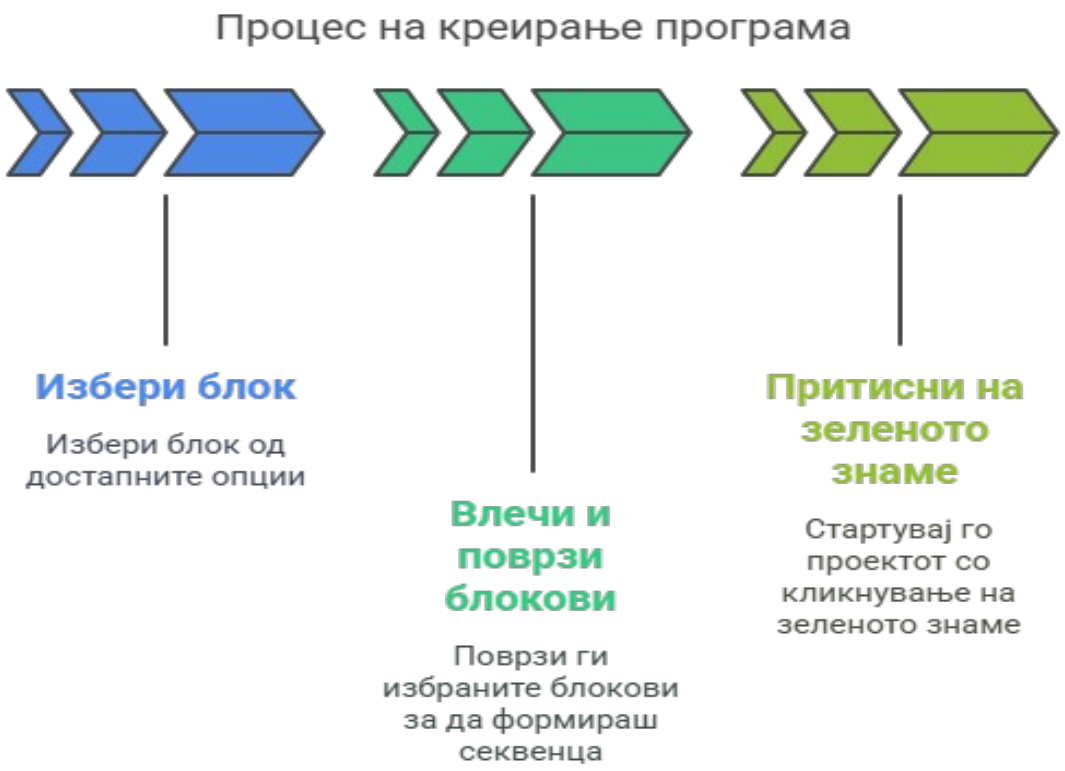
 Категории на наредби (со боја):

- **Движење (Motion)**
- **Изглед (Looks)**
- **Звук (Sound)**
- **Настани (Events)**
- **Контрола (Control)**
- **Молив (Pen)**
  ...и други.

 Креирање програма:

1. Избери блок
2. Влечи и поврзи блоковите
3. Притисни на **зеленото знаме** за да го стартуваш проектот

 Со Scratch, програмирањето станува **визуелно, лесно и забавно**, а учениците учат **логика, алгоритми и креативно решавање задачи.**


Процес на креирање програма
Избери блок
Избери блок од достапните опции
Влечи и поврзи блокови
Поврзи ги избраните блокови за да формираш секвенца
Притисни на зеленото знаме
Стартувај го проектот со кликнување на зеленото знаме

3.1.3 Креирање едноставни програми во Scratch

🎯 Проектна задача: Танцување со Cassy Dance

За подобро запознавање со Scratch, ќе направиме едноставна програма со **ликот Cassy Dance** кој ќе танцува на сцена.

🧩 Чекор 1: Стартување на програмата и избор на лик

- Отвори ја програмата Scratch.
- Избери го ликот **Cassy Dance**.
- Смести го во **средината на работната површина**.

Што е средина?
Средината е точката каде **координатите x и y се 0** → x = 0, y = 0

Работната површина е претставена како **координативен систем**, исто како во математика:

- **X-оска (хоризонтално)**
- **Y-оска (вертикално)**
- Центар = x: 0, y: 0

Чекор 2: Поставување на ликот во центар

- Отвори го менито **Motion**
- Избери ги блоковите:
  - set x to 0
  - set y to 0

Во местата каде внесуваш вредност (кружни полиња) – можеш **рачно да внесеш број**.
➡ Овие места се викаат **променливи (варијабли)**.

Чекор 3: Додавање воведна порака

- Отвори го менито **Looks**
- Избери блок:
  say [Порака] for [време] seconds

Вака ликот ќе поздрави и ќе изгледа како да се движи.

Чекор 4: Стартување на програмата

- Отвори го менито **Events**
- Додај блокот:
  when green flag clicked

 Овој блок означува **почеток на програмата**.

Чекор 5: Додавање позадина

- Кликни на иконата за **Backdrop**
- Избери позадина: Spotlight Stage 2
- Во делот **Backdrops**:
    - Десен клик на позадина → Duplicate
    - Со алатката **Fill with color** → обои ги новите позадини по желба

 Сите позадини можат да се менуваат, бојат и комбинираат.

Чекор 6: Промена на позадина преку код

- Отвори го менито **Looks**
- Додај блоковите за **менување позадини**
  switch backdrop to [...]

Чекор 7: Поставување објект на сцена

- Отвори го менито **Motion**
- Додај блокови:
    - set x to 15
    - set y to 100

 Сега ликот е поставен на точна позиција за изведба на танц.

🔹 Чекор 8: Додавање музика и движење

- Комбинирај блокови од **Sound**, **Motion** и **Looks**
- Додај го движењето на ликот и звучна поддршка (музика)

🔹 Чекор 9: Завршување на проектот

- На крајот на кодот, додај:
    - 🔹 say [Порака]
    - 🔹 stop all

🔹 Овие наредби го завршуваат проектот со поздрав и стопирање на сите активности.

▶ Старт и стоп

- **Зелено знаменце** – започнува проект
- **Црвено копче** – стопира проект

🔹 Запомни!

- Scratch користи **графички блокови** за програмирање.
- Блоковите се **боени** по категории.
- Работната површина е како **координативна мрежа**.
- **Променливи** се места за вредности.
- **Наредбите** му кажуваат на компјутерот **што да направи**.

Прашања

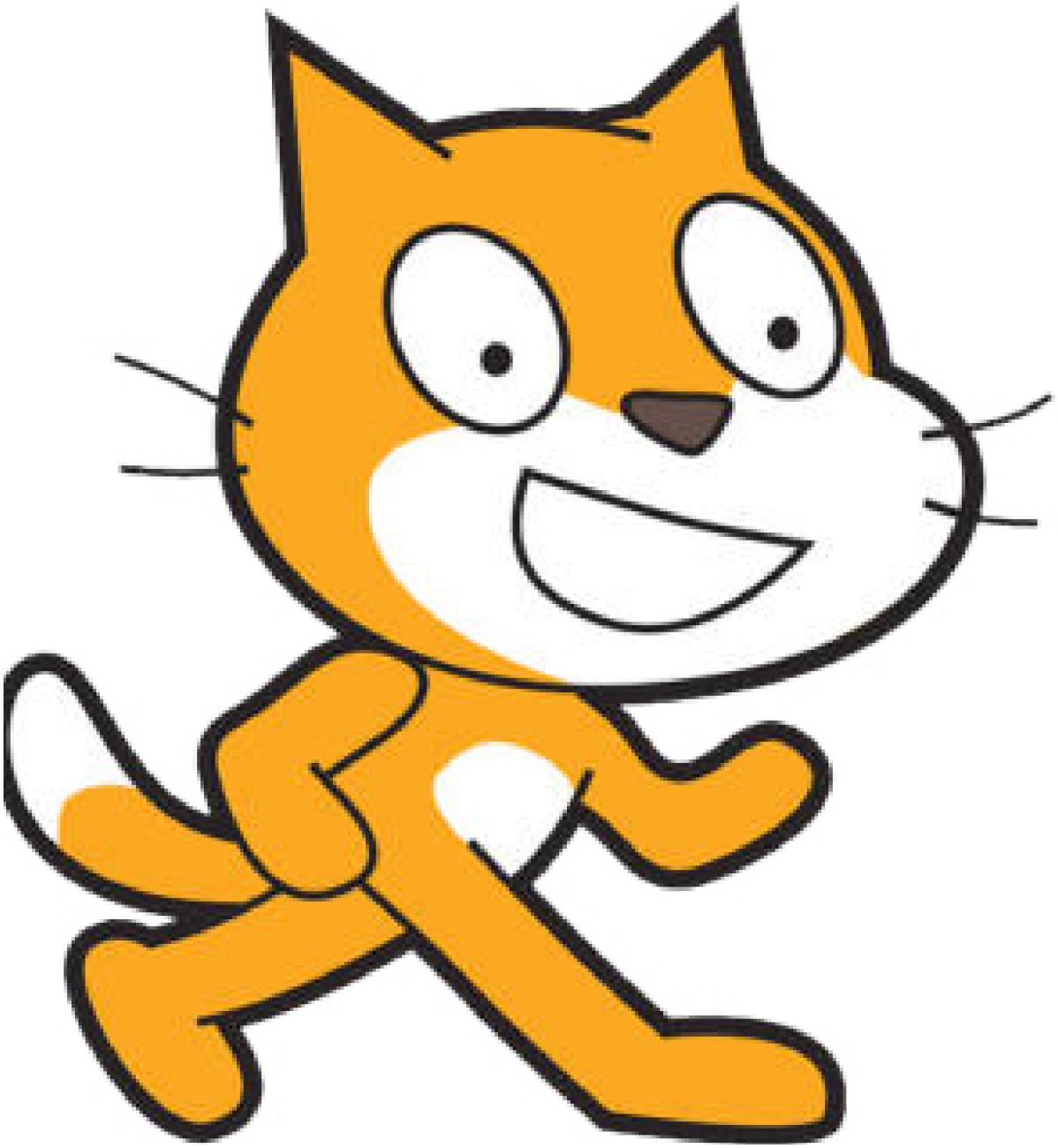
1. **Што означуваат блоковите во Scratch? На што наликуваат?**
   → Блоковите се наредби кои изгледаат како **делови од сложувалка**.

2. **Како се поделени блок-наредбите? Наведи неколку категории!**
   → Motion, Looks, Sound, Events, Control...

3. **Што означува зеленото знаменце кое се наоѓа во горниот дел на проектот?**
   → Тоа е **старт** на програмата.

4. **Што ни покажуваат координатите на објектот?**
   → **Точната позиција** на ликот на екранот.

5. **Како се поставува позадина?**
   → Со клик на иконата за позадини → избор или уредување на тема.



3.2 Интерактивни програми со настани

🔁 Да се потсетиме: Што значи „интерактивен“?

Терминот **интерактивен** значи **взаемна комуникација** или **размена на информации** помеѓу две страни.
Примери на интеракција:

- Час по информатика: наставникот и учениците комуницираат, поставуваат прашања, решаваат задачи.
- Образовен софтвер: програмата дава инструкции, ученикот одговара.
- Компјутерска игра: играчот игра според правилата, а играта реагира на неговите постапки.

🎯 Во оваа лекција ќе **креираме интерактивна програма** во Scratch со **настани**.

🎯 Проект: Математички калкулатор со Dinoaur1

Цел: Dinoaur1 ќе решава математички задачи (делење два броја) што ги внесува корисникот.

🧩 Чекор 1: Подготовка на проектот

1. Стартувај го **Scratch**
2. Избриши го ликот **Cat** со алатката ✂ *(scissors/delete)*
3. Додади нов лик: **Dinoaur1**
4. Додади **две позадини**:
    - Една креирана и обоена со алатките на Scratch
    - Една слика од компјутерот

Чекор 2: Почетна поставка

1. Кликни на ликот **Dinoaur1**
2. Преку менито **Motion**, постави го на почетна позиција:
    - set x to -122
    - set y to -85
3. Во **Events**, започни со:
    - when green flag clicked
4. Потоа, во **Looks и Motion**, додади:
    - say "Здраво математичари!"
    - glide 2 secs to x:0 y:0
    - Промена на боја
    - Промена на костим (за ефект на движење/размислување)

Чекор 3: Креирање на променливи (варијабли)

1. Во менито **Variables (Data)** → кликни **Make a variable**
2. Направи 3 променливи:
    - број1 (input од корисник)
    - број2 (втор број)
    - резултат (количник од број1 и број2)

⌨️ Чекор 4: Внесување вредности од тастатура

Користи блокови од категоријата **Sensing**:

- ask [Внеси го првиот број:] and wait
- set број1 to answer
- ask [Внеси го вториот број:] and wait
- set број2 to answer

🧮 Чекор 5: Пресметка на количник

- set резултат to број1 / број2
- say "Резултатот е [резултат]"

💡 Scratch автоматски ги претвора вредностите во броеви при математички операции.

🔁 Чекор 6: Целосна логика на програмата

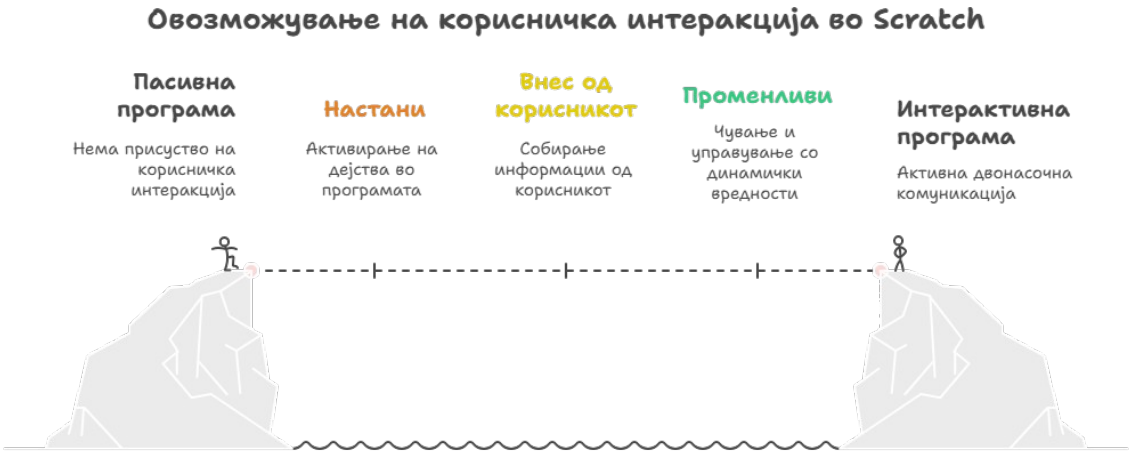
Редослед на блоковите:

1. when green flag clicked
2. go to x: -122 y: -85
3. say "Здраво математичари!"
4. glide to center
5. ask за број1
6. set број1
7. ask за број2
8. set број2
9. set резултат to број1 / број2

10. say "Резултатот е [резултат]"

Запомни!

- **Интеракција** е двонасочна комуникација: компјутер ↔ корисник.
- Во Scratch, интеракцијата се овозможува преку:
  - **настани** (Events) – почеток на дејство
  - **внес од корисникот** (Sensing → ask)
  - **променливи** за чување на вредности



❓ Прашања

1. **Што значи интерактивна програма?**
   → Програма која овозможува корисникот да влијае на нејзиното извршување преку внес или дејство.

2. **Кој е главниот лик во нашата интерактивна задача?**
   → Dinoaur1

3. **Кои променливи ги користиме во проектот?**
   → број1, број2, резултат

4. **Која е целта на проектот?**
   → Да се пресмета количник на два броја внесени од тастатура.

5. **Кој блок се користи за внесување вредност од корисникот?**
   → ask [прашање] and wait

🧩 Продолжение на проектот: Интерактивна пресметка во Scratch

⚠ Забелешка за именување на варијабли

Иако можеме на варијаблите да им дадеме **произволни имиња**, важно е тие **да бидат описни**, односно да ни покажуваат:

- каков тип на вредност содржат
- која е нивната улога во програмата

💡 Примери:

- деленик наместо број1
- делител наместо број2

- количник за резултатот од делението

Преглед на клучните чекори

---

1. Креирање на варијабли

Одиме во **Data → Make a variable**, и ги именуваме:

- деленик – вредност внесена од тастатура (прв број)
- делител – вредност внесена од тастатура (втор број)
- количник – резултат од деление

---

2. Почетна поставка

Преку менито Motion:

- go to x: -122 y: -85 → ја поставува почетната позиција на Dinoaur1

Преку Looks:

- switch backdrop to [почетна]
- say "Внеси го првиот број" → започнува интеракцијата

Преку Sensing:

- ask [Внеси го првиот број:] and wait
- set [деленик] to (answer)

➡ Истото го повторуваме за делител

---

3. Проверка дали делителот е 0

Користиме условна структура:

scratch

CopyEdit
if <(делител) = 0> then
  say "Делење со нула не е можно!"
  switch costume to [загрижен]
else
  set [количник] to (деленик / делител)
  say (join "Резултатот е: " количник)
  switch backdrop to [нова позадина]
  hide variable [деленик]
  hide variable [делител]
  hide variable [количник]
end

Дополнителна задача: Нови операции

➡ Креирајте блок-низи за пресметување на:

- **Собирање**
- **Одземање**
- **Множење**

Примери на нови варијабли:

- збир → за собирање
- разлика → за одземање
- производ → за множење

Пример: Собирање

scratch
CopyEdit
ask [Внеси го првиот број за собирање:] and wait
set [собир1] to (answer)

ask [Внеси го вториот број за собирање:] and wait
set [собир2] to (answer)

set [збир] to (собир1 + собир2)
say (join "Збирот е: " збир)

➡ Слично се прават блоковите за разлика и производ.

Завршување на проектот

Може да се додадат:

- Анимација на Dinoaur1 откако ќе се направи пресметката
- Преод во друга задача
- Звук по успешна операција

Практична задача за ученикот

Цел: Да се прошири Scratch проектот така што ќе може да:

- Пресметува збир, разлика, производ и количник
- Проверува дали има грешки (делење со 0)
- Менува позадина и изглед на Dinoaur1 во зависност од ситуацијата

Размисли!

Дали може да се креира низа од блок-наредби кои ќе вршат пресметки на математички операции со три броја?

Да, може!
Наместо само два броја (`број1` и `број2`), можеме да креираме и **трет број** – на пример `број3`.
Потоа можеме да направиме пресметка од типот:

(број1 + број2 + број3)
((број1 * број2) - број3)
((број1 + број2) / број3) и сл.

Што ни недостига во програмот на Dinoaur1?

Во оригиналниот проект:

- Има **само два броја**

- Пресметува **едноставна операција** (делење)

За да додадеме работа со три броја, ни треба:

1. **Уште една варијабла** → на пример: broj3
2. **Дополнителни ask и set блокови** за внесување вредност
3. **Промена во блоковите каде се прави пресметката** – сега ќе користиме три вредности наместо две.

Каде би направил измени?

Првиот дел – внес на вредности:

```
scratch
CopyEdit
ask [Внеси го првиот број:] and wait
set [број1] to (answer)

ask [Внеси го вториот број:] and wait
set [број2] to (answer)

ask [Внеси го третиот број:] and wait
set [број3] to (answer)
```

Пресметка (пример за збир на три броја):

```
scratch
CopyEdit
set [збир] to ((број1 + број2) + број3)
say (join "Збирот е: " збир)
```

ПРАШАЊА

1. Кои програми се нарекуваат интерактивни?
2. Што се варијабли? Која е постапката за креирање варијабли?
3. Со која наредба се овозможува внесување вредност на варијаблата?
4. Како корисникот внесува вредност на варијаблата во програмата?
5. Зошто служи наредбата If...Else?

🎮 3.3 Изработка на програми со посложени проблемски ситуации – Scratch игра: Cat и Gobo

---

🎯 Цел на играта:

Играчот (Cat) да ги фаќа паѓачките Gobo подароци. Секој фатен Gobo дава 1 поен. Играта може да има победнички услов (на пример: 10 поени = победа).

🧩 ЧЕКОРИ ЗА КРЕИРАЊЕ НА ИГРАТА

---

1. 🐱 Додавање на објекти (Sprites):

- Додади: Cat и Gobo
- Намали ги со **алатката Shrink** за да одговараат на просторот
- Додај **позадина**: Gingerbread или друга по избор

---

2. 🐱 Програмирање на Cat (Sprite1)

🎯 Цел: Cat да се движи лево-десно според глувчето

```
scratch
CopyEdit
when green flag clicked
go to x: 0 y: -130
point in direction 90
forever
  go to x: (mouse x) y: -130
end
```

🔊 Звук на почетокот:

```
scratch
CopyEdit
when green flag clicked
play sound [dance slow mo v] until done
```

---

3. 🍪 Креирање клонови на Gobo што паѓаат од небото

🎯 Цел: Да се создаваат нови Gobo на секунда

```
scratch
CopyEdit
when green flag clicked
forever
  create clone of [myself v]
  wait (1) seconds
end
```

---

4. 🍪 Инструкции за секој клон на Gobo

🎯 Цел: Клоновите паѓаат од горе и исчезнуваат кога ќе го допрат Cat

```
scratch
CopyEdit
when I start as a clone
go to x: (pick random -200 to 200) y: 140
repeat until <touching [Cat v]>
  change y by -5
  wait (0.05) seconds
end
play sound [zoop v]
change [поени v] by (1)
delete this clone
```

---

5. 🔢 Креирање променлива поени за броење

🎯 Постапка:

1. Кликни на Data → Make a variable

2. Име: поени (или Points)
3. За сите спрајтови
4. Додај:

```
scratch
CopyEdit
when green flag clicked
set [поени v] to (0)
```

---

6. 🔁 Додавање услов за победа (опционално)

```
scratch
CopyEdit
if <(поени) = 10> then
  say [ПОБЕДА!] for (2) seconds
  stop all
end
```

🔁 Може да го ставиш ова **внатре во forever циклусот кај Cat**, за да го следи резултатот.

Категорија	Блок	Цел
Events	when green flag clicked	Почеток на играта
Motion	go to x: y: / change y by	Движење на Cat и Gobo
Control	forever, repeat until, if...then	Циклична логика
Sensing	touching [Cat v]?	Судир
Sound	play sound [zoop v]	Ефекти при фаќање
Data	set, change, make variable	Броење поени
Looks	say [текст] for n seconds	Пораки за победа и сл.

🧠 ЗА РАЗМИСЛУВАЊЕ: Модификации и проширувања

- Додај **тајмер** – да се игра 30 секунди

- Додај **животи** – ако Gobo падне до крај, намалуваш живот
- Додај **нови нивоа** – побрзо паѓање со секои 5 поени
- Додај **анимации или звучни ефекти** на победа/пораз

Резиме

Овој проект во Scratch:

- вклучува **движење, клонови, интеракции, поени и звук**
- е **интерактивна програма**, бидејќи реагира на дејствија на корисникот
- користи **променливи, услови и циклуси** за динамичка логика
- е основа за создавање **повеќениво игра** со лесно проширување

Задача

Креирајте едноставна игра во Scratch. Игрите кои ги креиравте споделете ги со другарчињата и играјте следејќи ги инструкциите.

Запомни!

Проектите кои вклучуваат повеќе објекти и повеќе настани се посложени за креирање. Секој објект посебно се програмира, односно му се задаваат низи од блок-наредби за извршување со цел создавање движење, активност или дејство. Во програмите објектите се опишуваат со состојба и начин на однесување. Во реалниот свет објект може да биде: автомобил, велосипед, топка или некој лик кој е дефиниран со состојба и однесување. Настаните се случуваат зависно од состојбата или однесувањето на објектот. Тие треба да се предвидат и за нив да се понуди соодветно решение

Прашања

1.Што се објекти?

2.Како се програмираат објектите?

3.Која е врската меѓу објектите и настаните?

Програмирање преку стандарден структурен програмски јазик (C++)

1. Вовед во програмирање

- Што е програмски јазик?
- Зошто го учиме C++?

2. Процес на изработка на програма

- Идеја → Планирање → Кодирање → Тестирање → Поправки

3. Интегрирана околина за програмирање (IDE)

- Запознавање со Code::Blocks

- Алати и функции во IDE

🧑‍💻 4. Примери на програмски кодови

- Прв „Hello World“ пример
- Анализа на делови од кодот

▶ 5. Извршување на пример програми

- Чекори за компајлирање и извршување
- Препознавање на грешки

🧱 6. Основни елементи на C++

- main() функција
- Знаци за коментари (// и /* */)
- Блок структура { }

📝 7. Искази (Statements)

- Наредби за извршување
- Редослед на извршување

✍ 8. Изработка на свои програми

- Како да планираш и напишеш едноставна програма
- Корисни совети

🔢 9. Аритметички операции и изрази

- Значење на +, -, *, /, %
- Редослед на извршување

📦 10. Константи и променливи

- Што е променлива?
- Како се декларираат и користат?

📥 11. Внесување податоци

- Користење на cin за внес
- Примери со текст и бројки

🔍 12. Споредбени изрази

- Условни проверки со ==, !=, <, >, <=, >=

- Користење во if изрази

🔁 13. Повторување до исполнување на услов

- Циклус while и do...while
- Примери од реалниот живот (калкулатор, броење)

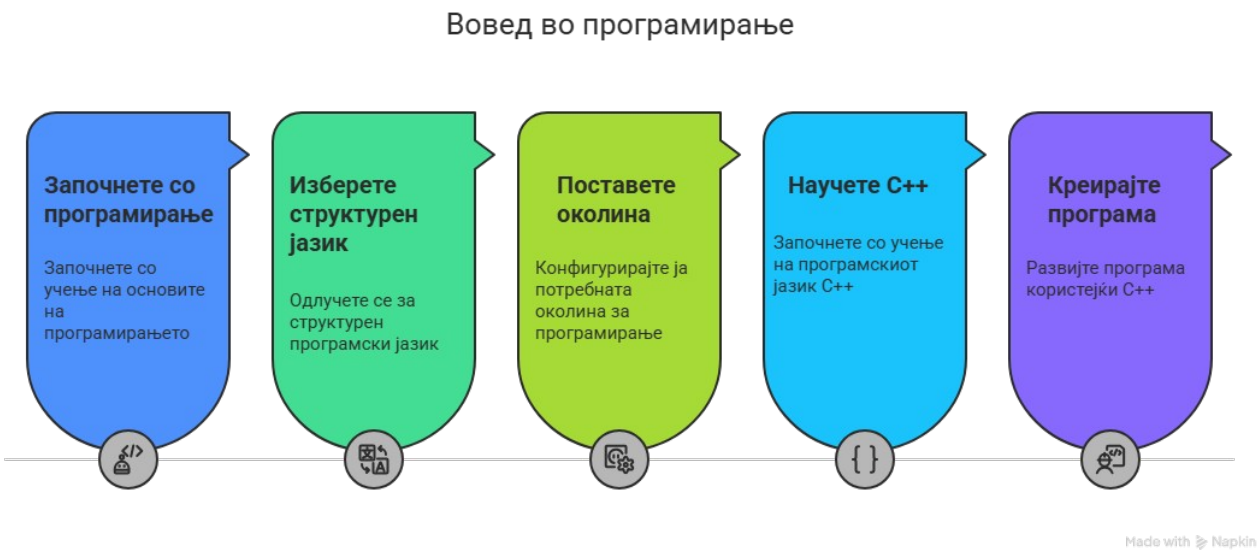
🧪 14. Да повториме! Да увежбаме!

- Прашања за повторување
- Задачи за самостојна работа
- Мини-проект

Поглавје 4: Вовед во програмирање преку стандарден структурен програмски јазик (C++)

🎯 Што ќе научиме во ова поглавје?

🔹 Да работиме во **интегрирана околина за програмирање** — **Code::Blocks**
🔹 Да напишеме и извршиме своја прва програма на **јазикот C++**
🔹 Да ги разбереме **основните делови од една програма**
🔹 Да се запознаеме со **променливи, изрази и аритметика** во C++
🔹 Да започнеме со **внесување и обработка на податоци**


Вовед во програмирање
Започнете со програмирање
Започнете со учење на основите на програмирањето
Изберете структурен јазик
Одлучете се за структурен програмски јазик
Поставете околина
Конфигурирајте ја потребната околина за програмирање
Научете C++
Започнете со учење на програмскиот јазик C++
Креирајте програма
Развијте програма користејќи C++
Made with Napkin

□□ Што е тоа **програмирање**?

Програмирање е **пишување на упатства (инструкции)** кои компјутерот ги извршува. Овие упатства се нарекуваат **програми**, а ги пишуваме во **програмски јазици** како што е **C++**.

□ Зошто користиме **C++**?

- Тоа е еден од најмоќните и најкористени програмски јазици
- Се користи во игри, роботи, апликации, софтвер за машини
- Ја учиме **структурата на програмите** и **логиката на решавање проблеми**

□ Што е **интегрирана развојна околина (IDE)?**

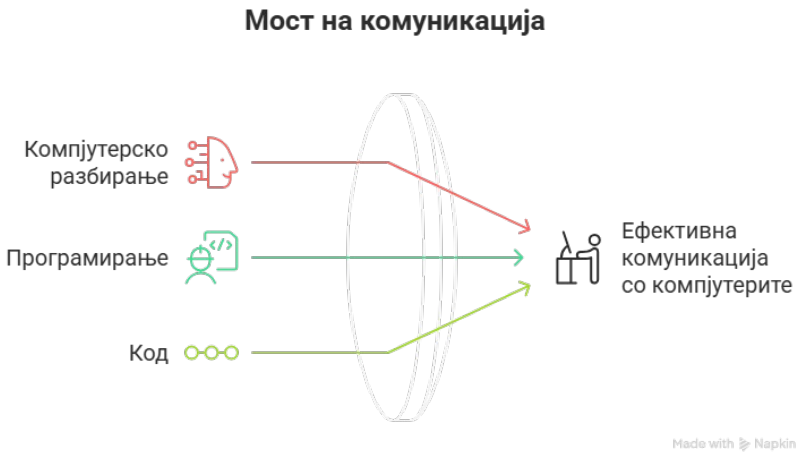
Code::Blocks е бесплатна програма која ни овозможува:

- да пишуваме код ✍
- да го проверуваме за грешки □
- и да го извршуваме

Визуелни елементи за страницава:

(овие може да ги изгенерираш со Napkin.ai или додадеш во Canva)

- 🖼️ Слика од Code::Blocks интерфејс
- *Компјутерите не разбираат луѓе – затоа ние учиме да зборуваме со нив преку код!“*



- 🖼️ Мал мини-квиз (пример: Што е програма? а) Видео игра б) Низ инструкции за компјутер в) Музика)

Добредојде во светот на програмирањето!

🌍 Живееме во време на брзи промени

Технологијата е насекаде околу нас – во училиштата, болниците, продавниците, па дури и во нашите џебови преку мобилните телефони. 🖼️

🤔 Што е компјутер?

Компјутерот е **електронски уред** кој може да:

- 📥 *прибира податоци*
- 💾 *ги чува податоците*
- ⚙️ *ги обработува податоците*
  ...и сето тоа го прави со помош на **програми**!

🤔 Што се програми?

Програмите се инструкции кои му кажуваат на компјутерот што да прави. Без програми, компјутерот не може да прави ништо корисно.

🎮 Примери од секојдневието:

🎮 Играме игри
📝 Пишуваме во Word или Writer
📊 Пресметуваме во Excel или Calc
📽️ Правиме презентации во PowerPoint или Impress
🎬 Гледаме видеа во VLC или RealPlayer
🎨 Цртаме во Paint или Photoshop
💬 Разговараме на Viber, WhatsApp и Messenger

👨‍💻 Кој ги прави програмите?

➡ **Програмерите** се луѓе кои создаваат програми.
➡ Тие користат **програмски јазици** како **C++, Python, Java и други.**

📚 Што се програмски јазици?

Програмски јазик е **специјален начин на пишување инструкции** што компјутерот ги разбира. Секој јазик има:

- 🔣 Симболи
- 🧠 Клучни зборови
- 📏 Правила на пишување (синтакса)

🚀 Зошто да учиме програмирање?

🧠 Го развива **логичкото размислување**
🔍 Помага во **решавање проблеми**
💪 Го тренира мозокот како гимнастика!
🌟 Може да стане и твојата идна професија!

📌 Да запомниме:

> 💡 **Програмирањето** е креативен процес на решавање проблеми со помош на код.

 Ти можеш да станеш креатор на софтвер, игри, апликации и многу повеќе!

Клучни поими:

Поим	Објаснување
Преведувач (Compiler)	Претвора код во јазик што компјутерот го разбира
Изворна програма	Кодот што го пишува програмерот
Извршна програма	Програмата што се стартува на компјутерот
✍ Едитор	Програмата во која се пишува кодот (на пр. Code::Blocks)
Дебагер	Алатка за пронаоѓање грешки
Исказ	Наредба во програмата
Споредбен израз	Проверка: дали два броја се еднакви?
Циклус	Код што се повторува додека не се исполни услов

- Првиот програмер била Ада Лавлејс – уште во 1800-тите!

-Ада Лавлејс

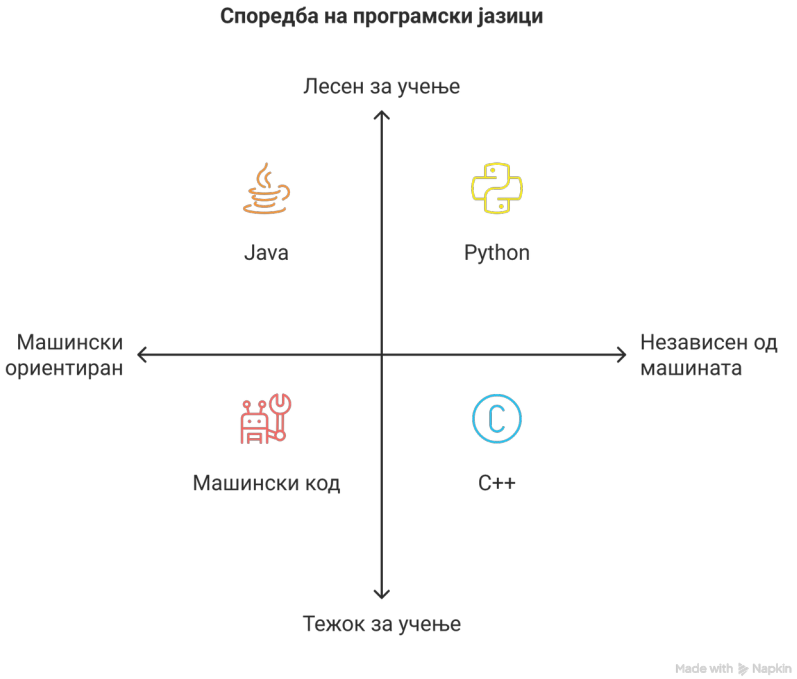
Процес на изработка на една програма

🔹 Што е програма?

Програма е **серија на упатства** (инструкции) напишани на **програмски јазик**, кои компјутерот ги извршува за да реши некоја задача.

Платформи каде се користат програми:

- 🔹 Компјутери
- 🔹 Мобилни телефони
- 🔹 Таблети
- 🔹 Вградени уреди (како паметни фрижидери, часовници и др.)



Програмски јазици: Виши vs. Нижи

Вид на јазик	Објаснување	Пример
Нижи програмски јазик	Машински ориентиран, зависи од конкретен процесор. Тежок за учење.	Машински код, Асемблер
Виши програмски	Независен од машината. Полесен за разбирање. Сличен на природен јазик.	C, C++, Java, Python

Вид на јазик	Објаснување	Пример јазик

 Вишите програмски јазици се најчесто користени денес!

Фазите при креирање програма

Без разлика кој јазик го користиме, процесот на изработка на програма обично вклучува:

Процес на развој на програма во C++
Вовед во Code::Blocks
Запознавање со интегрираната околина за програмирање
Конфигурирање на Code::Blocks за развој на C++
Поставување на околината
NEW
Креирање на нов проект
Иницијализирање на нов проект во Code::Blocks
Пишување на C++ код во уредувачот
Пишување на кодот
Компилација на кодот
Преведување на кодот во извршна форма
Стартување на програмата за да се види резултатот
Извршување на програмата
Made with Napkin

Табела 1: **Фази при пишување програма**

Фаза	Опис
1. Дефинирање на проблемот	Што треба да направи програмата?
2. Осмислување решение	Избирање на логика и чекори
3. Пишување на код (кодирање)	Се користи програмски јазик
4. Тестирање и отстранување грешки	Се користи дебагер
5. Завршна проверка и користење	Подготвена за корисникот

🔚 Краток заклучок:

- Програмите се создаваат со **програмски јазици**
- Постојат **нижи** и **виши** јазици
- **C++** е еден од најпознатите виши јазици – и токму него ќе го учиме!

🧠 Мини-квиз (долен блок)

Прашање:
Koj jазик е полесен за разбирање?
а) Ниж
б) Виш
в) Машински код

Точен одговор: б)

Слики што можеш да додадеш (Canva/Napkin.ai):

- Дијаграм: „Фази на програмирање“
- Икони за кодирање, тестирање, процесори
- Слика со споредба: 0s и 1s (машински код) vs. cout << "Здраво!" (виш код)

Обидете се! – Програмирање во Code::Blocks

Ќе работиме со:

- Програмски јазик: **C++**
- Развојна околина: **Code::Blocks**

Потсетување:

- *Програмите се низи од инструкции кои компјутерот ги извршува.*
- *Процесот се вика програмирање, а лицето што ги пишува се нарекува програмер.*

Кратки факти:

Поим	Објаснување
Програмирање	Процесот на создавање програми
Програмер	Лице што пишува код
Изворен код	Напишаната програма (филесо наставка .cpp)

Поим	Објаснување
Извршна програма	Програмата што ја извршува компјутерот
Синтакса	Начинот на кој се пишуваат командите во јазикот
Програмски јазици	Делат се на **нижи** (блиски до машината) и **виши** (разбирливи за луѓето)

Фази при создавање програма:

1. ✍ Пишување изворен код
2. Преведување (компајлирање)
3. Поврзување во извршна програма
4. Тестирање
5. Верификација (дали сè работи како што треба)

Да провериме што научивме! (Прашања со простор за одговори)

1. Како се вика постапката за креирање програми?

_____

2. Кој ги креира програмите? Како?

_____

3. Како се делат програмските јазици?

_____

4. Кои се нижи програмски јазици?

_____

5. Наброј некои виши програмски јазици:

_____

6. Која е разликата помеѓу виши и нижи јазици?

_____

7. Што е синтакса на програмски јазик?

_____

8. Што е изворна програма? Каква наставка има?

_____

9. Како се вика програмата што ја извршува компјутерот?

_____

10. Набројги фазите при креирање програма:

_____

4.2 Запознавање со основните елементи на интегрирана околина за програмирање

Што е IDE (Integrated Development Environment)?

IDE е пакет со алатки што им помага на програмерите да:

Пишуваат код
Проверуваат за грешки
Преведуваат код (компајлирање)
Тестираат програма
Уредуваат и подобруваат софтвер

Зошто се користи IDE?

➡ За да биде **сè на едно место**
➡ За да се **намалат грешките**
➡ За да се **забрза програмирањето**

 IDE што ќе го

Основни елементи на едно IDE
Уредник
Област каде се пишува кодот.
Компилатор
Го преведува кодот во јазик што го разбира компјутерот.
Дебагер
Ги пронаоѓа и решава грешките во кодот.
Библиотеки
Подготвени делови од код што можеме да ги користиме.
Алатка за извршување/ тестирање
Алатка за извршување и тестирање на програмата.
Made with Napkin

користиме: **Code::Blocks**

- Бесплатен и отворен код
- Работи со **C++** и други програмски јазици
- Вклучува: едитор, компајлер, дебагер
- Работи на повеќе оперативни системи (Windows, Linux...)

Обидете се! – Размислувај како IDE!

Задача:
Замисли дека ти си IDE и треба да помогнеш на другарче во училиште:

1. **Креирај „чекори“ (инструкции)** како да реши проблем (на пример: како да се подготви за тест).
2. **Преведи ги** тие чекори на јазик што сите во одделението ќе го разберат.
3. **Провери** дали решението функционира и предложи подобрувања.

Мини-квиз

1. Што е IDE?

_____

2. Кои алатки содржи едно IDE?

_____

3. Како се вика IDE-то што го користиме во C++?

 Визуелен додаток (можеш да ставиш во Canva или Writer):

- Дијаграм „IDE – All-in-one tool“ со икони (Editor, Compiler, Debugger)
- Слика од интерфејсот на Code::Blocks
- Мала илустрација: „Јас сум IDE!“

4.2.1 Инсталирање на Code::Blocks

1. Преземање на програмата

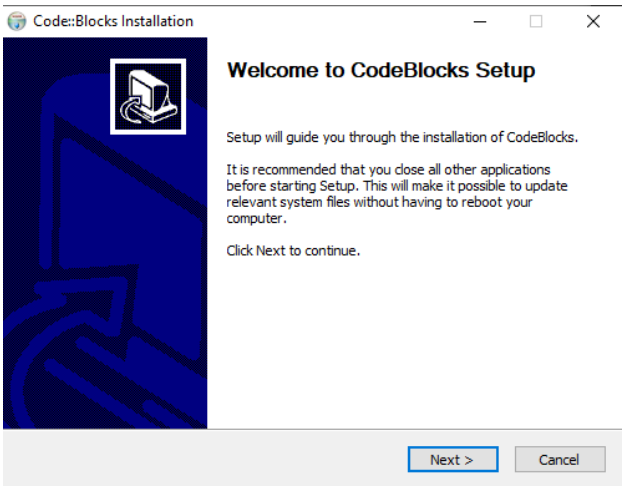
➡ **Code::Blocks** е бесплатна програма со отворен код.
 Може да се преземе од:
 https://www.codeblocks.org/downloads/binaries

⬇ Преземи ја верзијата за Windows (пример: codeblocks-20.03mingw-setup.exe)

2. Почеток на инсталација

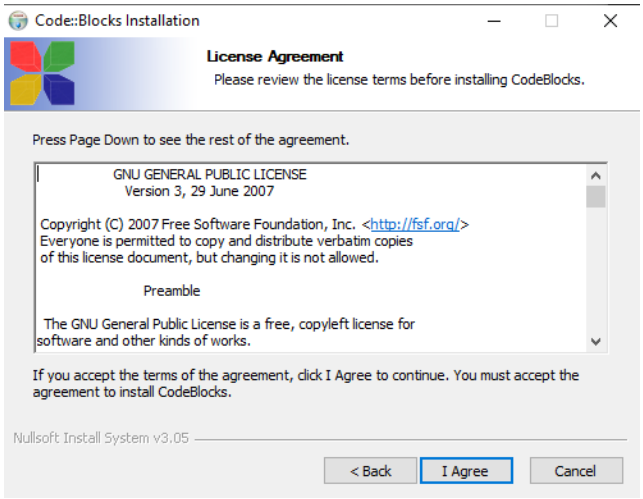
 Двапати кликни на преземениот фајл
♂ Се појавува **инсталацискиот волшебник (Wizard)**

 Кликни на копчето Next



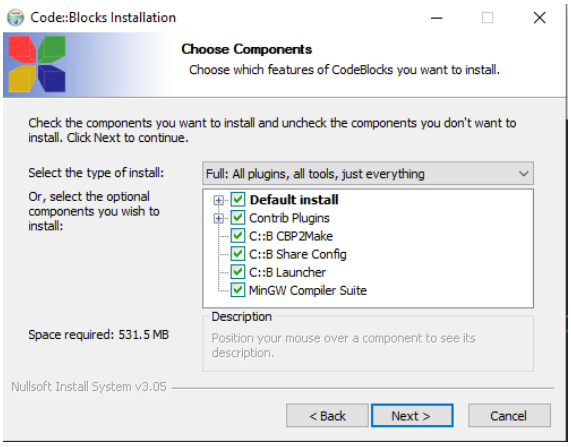
3. Прифаќање на лиценцата

 Се појавува прозорец со правилата за користење
☑ Кликни на I Agree за да продолжиш



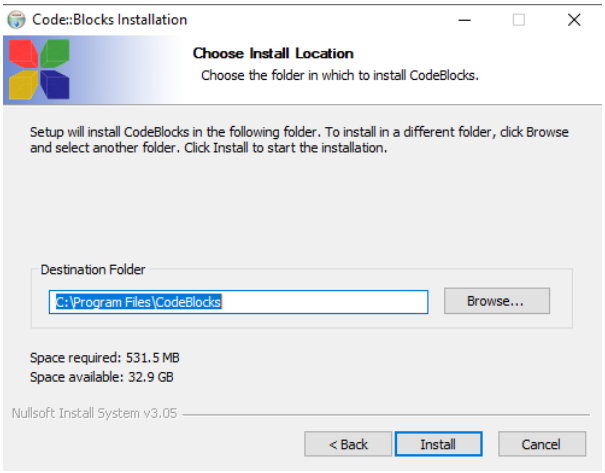
⚙ 4. Избор на компоненти

 Избери што да се инсталира (остави ги стандардните опции)
 Кликни Next



5. Избор на дестинациски фолдер

 Избери каде ќе се инсталира програмата (на пр. C:\Program Files)
 Кликни Next

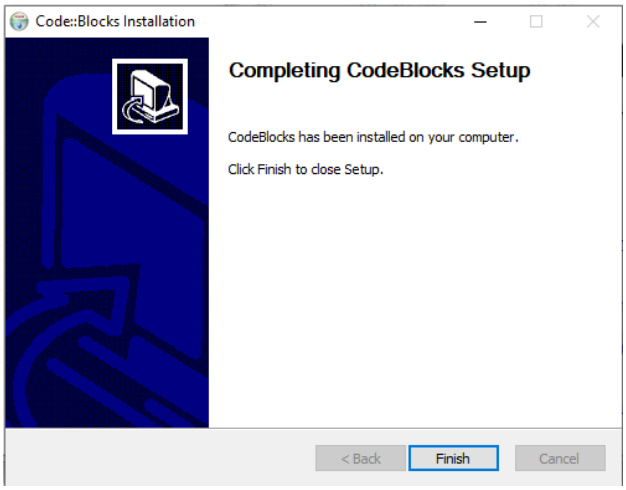


6. Инсталација

 Кликни Install – се копираат фајловите
 Чекај додека не заврши процесот

7. Завршување

По инсталацијата, кликни Next, а потоа Finish



8. Стартување на Code::Blocks

Се појавува прозорец со прашање дали да се стартува Code::Blocks
Избери Yes
➡ Ќе се отвори **работната површина** на интегрираната околина

4.2.2 Работна околина на Code::Blocks

Креирање нов проект

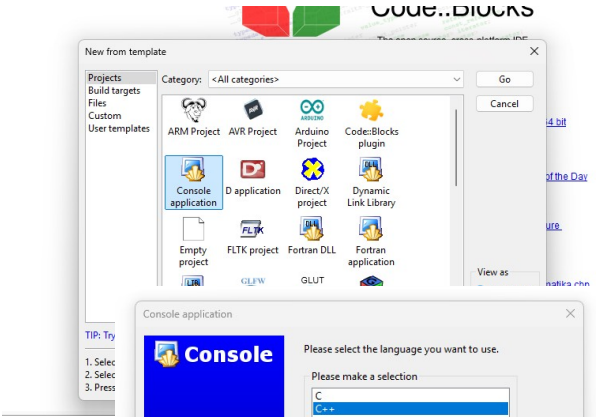
Чекор 1– Нов проект

Кликни на:
File → New → Project

Слика 8: Креирање нов проект

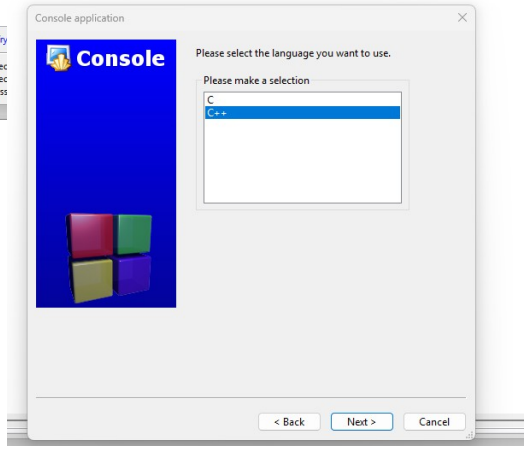
Чекор 2 – Избор на тип

Избери: **Console Application**
Кликни на Go

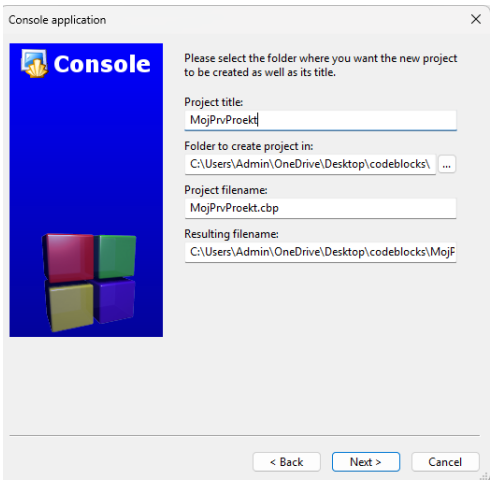


Чекор 3 – Избор на програмски јазик

Избери: **C++**
Кликни Next



Чекор 4 – Доделување име и локација



 Внеси:

- Име на проект: MojPrvProekt
- Избери каде ќе биде зачуван (на пример: Desktop/Проекти)

 Кликни Next

Чекор 5 – Избор на компајлер

 Остави го предложениот компајлер (обично е GNU GCC Compiler)

 Кликни Finish

Резултат:

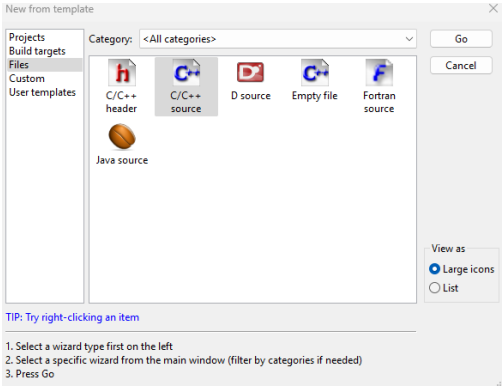
 Сега си спремен да започнеш со пишување изворен код во едиторот на Code::Blocks!

4.2.3 Креирање нов фајл за изворен код во Code::Blocks

Чекор по чекор: Креирање на .cpp фајл

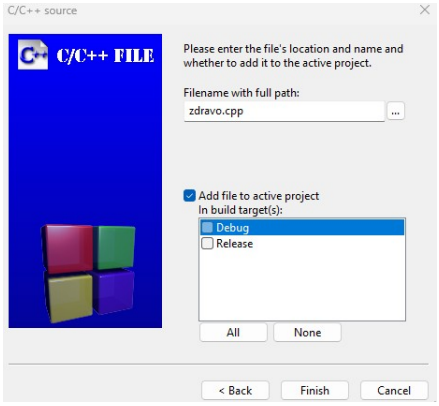
1. **Мени**:
File → New → File

2. **Избор на тип на фајл**:
 Избери: C/C++ Source
 Кликни Go



3. **Избор на програмски јазик**:
 Одбери: C++
 Кликни Next

4. **Име и локација на фајлот**:
 Кликни Browse → избери фолдер
 Внеси име, на пример: zdravo.cpp
 Кликни Finish



✍ Совет за учениците

 На твојот десктоп креирај посебна папка, на пример: **„C++Проекти“**, каде ќе ги зачувуваш сите изворни фајлови.

Што е Интегрирана Околина за Програмирање (IDE)?

 Тоа е софтвер кој ги содржи сите потребни алатки за програмирање:

- ✍ **Едитор** за пишување код
- **Компајлер** за превод на код
- **Дебагер** за откривање грешки
- **Библиотеки и алатки** за полесно програмирање

➡ Code::Blocks е токму таква бесплатна интегрирана околина со отворен код.

Запомни! (Прашања за проверка на знаење)

Прашање	✍ **Одговор**
1. Што е интегрирана околина за	

❓ **Прашање**	✍ **Одговор**

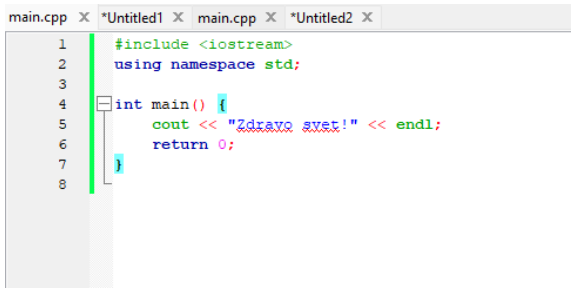
програмирање?
2. Која е нејзината главна намена?
3. Кои алатки ги содржи?
4. Каде се пишува изворниот код?
5. Што прави компајлерот?
6. Што прави дебагерот?
7. Опиши ја работната површина на Code::Blocks.

4.3 Изглед на готови пример-програмски кодови

✍ Што е изворен код?

🔹 **Изворен код (source code)** е кодот што го пишуваме во **едитор** со цел да создадеме програма.
🔹 Тој мора да биде напишан со **точна синтакса и логика (семантика)** според правилата на програмскиот јазик.

🔹 Пример на C++ код:



🔹 *Секоја линија има број*
🔹 *Секоја наредба завршува со* ;
🔹 *Грешките се прикажуваат долу во Build log*

🔹 Совет!

> 🔹 За поголема читливост, **разделувај ги наредбите по линии**, дури и ако не е задолжително.
> Преведувачот (компајлерот) ги разбира наредбите според ;, но програмерите – по редослед и структура!

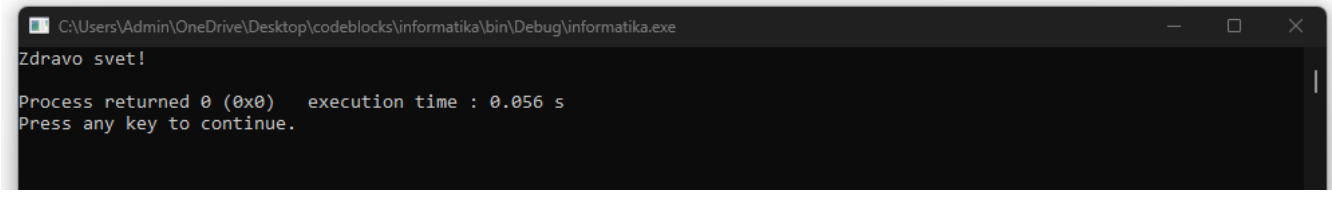
⚙ Како се преведува код?

 Преведување = Build
 Извршување = Run

Акција	Начин	Икона или кратенка
Преведи код	Build → Build	⌨ Ctrl + F9
Изврши програмата	Build → Run	⌨ Ctrl + F10
Резултат	Појавување на нов прозорец со извршен код	

 Ако има **грешка**, се појавува долу во прозорецот **Build log**
 По успешно извршување се создаваат **три типа на фајлови**:

Прозорец на извршена програма



 По пораката, корисникот треба да притисне било кое копче за прозорецот да се затвори.

Запомни!

Прашање	✍ Одговор (ученикот пополнува)
1. Што е изворен код?	
2. Со која наредба се преведува изворен фајл?	
3. Кои фајлови се создаваат по компајлирање?	
4. Како се извршува програма во Code::Blocks?	

4.4 Извршување на готови пример-програми

Што ќе научиме?

Во оваа лекција ќе:

- користиме **внес од тастатура** во програма,
- **пресметаме збир или производ** на два броја,
- **откриеме и поправиме грешки (bugs)**,
- научиме како **да дебагираме**,
- и **да го извршиме** финалниот код.

Пример: Програма за збир на два броја

```
1  // Оваа програма пресметува збир на два цели броја
2  #include <iostream>
3  using namespace std;
4
5  int main() {
6      int br1, br2, zbir;
7
8      cout << "Vnesi go prviot broj: " << endl;
9      cin >> br1;
10     cout << "Vnesi go vtoriot broj: " << endl;
11     cin >> br2;
12
13     zbir = br1 + br2;
14
15     cout << "Zbirot e: " << zbir << endl;
16     return 0;
17 }
18
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\informatika\bin\Debug\informatika.exe
Vnesi go prviot broj:
2
Vnesi go vtoriot broj:
2
Zbirot e: 4

Process returned 0 (0x0)   execution time : 8.209 s
Press any key to continue.
```

Клучни објаснувања:

Линија	Објаснување
//	Коментар (не се извршува)

Линија	Објаснување
#include <iostream>	Вклучување стандардна библиотека
main()	Главната функција на секоја програма
int br1, br2, zbir;	Дефинирање променливи
cin >>	Внес од тастатура
cout <<	Испис на екран
zbir = br1 + br2;	Пресметка
return 0;	Завршување на програмата без грешка

⚙ Процес на извршување

1. **Преведување (Build):**
   Build → Build или Ctrl + F9
2. **▶ Извршување (Run):**
   Build → Run или Ctrl + F10
3. **Се добиваат фајлови:**
   - *.cpp → изворен код
   - *.obj / *.o → објектен фајл
   - *.exe → извршен фајл

Што ако има грешка?

Пример за погрешен код:

main.cpp

```
1   #include <iostream>
2   using namespace std;
3
4   int main() {
5       int br1, br2, Proizvod
6       cout << "Vnesi broevi: ";
7       cin >> br1 >> br2;
8       Proizvod = br1 * br2
9       system("PAUSE");
10      return 0;
11  }
12
```

Logs & others

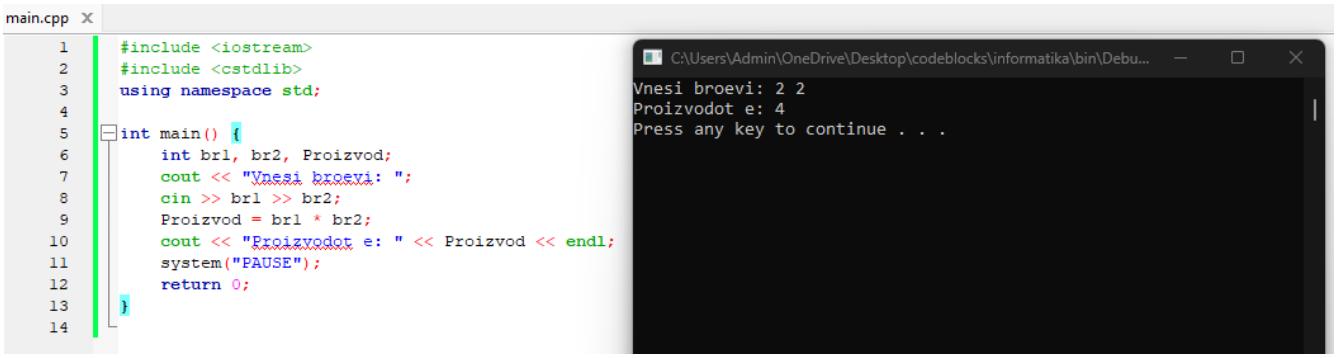
Code::Blocks | Search results | Cccc | Build log | Build messages | CppCheck/Vera++ | CppCheck/Vera++ m

File	Line	Message
		=== Build: Debug in informatika (compiler: GNU GCC Compiler) ===
C:\Users\Admi...		In function 'int main()':
C:\Users\Admi...	6	error: expected initializer before 'cout'
C:\Users\Admi...	8	error: 'Proizvod' was not declared in this scope
		=== Build failed: 2 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===

Грешки:

Линија	Грешка
int br1, br2, Proizvod	Нема ;
Proizvod = br1 * br2	Нема ;
system("PAUSE")	Недостасува #include <cstdlib>

Поправен код:



Запомни!

Прашање	Одговор
1. Како започнува секоја C++ програма?	Со функцијата int main()
2. Дефинирај вовед во програма!	Делот каде се вклучуваат библиотеки
3. Што означуваат //?	Коментари во кодот
4. Стандардна библиотека?	<iostream>
5. Како се вика главната функција?	main()
6. Како се викаат грешките?	Bugs
7. Како се отстрануваат?	Со дебагирање (Debug)
8. Што значи return 0;?	Програмата завршила успешно
9. Со кој знак се одвојуваат	Со ; (точка и запирка)

Прашање **Одговор**

наредбите?

4.5 Основни елементи на програмскиот јазик C++

Методи на комуникација со компјутерот
Влезни уреди
Корисникот комуницира со компјутерот преку влезни уреди како што се тастатура или маус.
10010
01100
Превод на бинарен код
Податоците од влезните уреди се преведуваат во бинарен код за компјутерот да разбере.
Комуникација преку мрежи
Компјутерите комуницираат меѓусебно преку мрежи, овозможувајќи размена на податоци.
Made with Napkin

Азбука на програмскиот јазик:

Се состои од:

- **Букви (големи и мали)** → A-Z, a-z
- **Цифри** → 0-9
- **Специјални знаци** → +, -, *, /, =, ;, (), {}
- **Празно место**
- **Комбинации од знаци**

Градбени елементи на C++:

Елемент	Опис	Пример
Резервирани зборови	Веќе дефинирани зборови	int, if, while, true, else
Идентификатори	Имиња што ги дава програмерот	zbir, broj1, _rezultat
Оператори	Симболи за операции	+, -, *, /, ==, !=
⚠ Интерпункциски знаци	За структура на кодот	;, (), {}
Коментари	Објаснувања во кодот	// пресметка на збир

⚙ Правила за идентификатори:

Може:

- Да почнуваат со буква или долна црта
- Да содржат букви, цифри и долна црта

Не смеат:

- Да почнуваат со цифра
- Да содржат специјални знаци (!, @, #)
- Да бидат исто како резервирани зборови

Примери на исправни идентификатори:

- zbir, broj1, _max, rezultat_2025

Основни елементи во C++

Елемент	Опис	Пример
Податоци	Информации со тип, име, вредност	int, float, bool
Константи	Фиксни вредности	const int PI = 3;
Променливи	Можат да се менуваат	int broj = 5;

Елемент	Опис	Пример
Декларации	Опишуваат променливи	float a;
Изрази	Пресметки	zbir = a + b;
Искази	Наредби за извршување	if (a > b)
Функции	Група на искази	main()
Модули	Делови од програма во боја	(во IDE, боја за int, cout, итн.)

Структура на C++ програма:

main.cpp

```cpp
#include <iostream>   // библиотека
using namespace std;  // простор на имиња

int main() {          // главна функција
    // коментар
    int a = 5;        // декларација и иницијализација
    cout << a;        // исказ
    return 0;
}
```

Запомни! –

Одговори

1. **Наброј ги градбените елементи на програмскиот јазик C++:**
   - Резервирани зборови
   - Идентификатори
   - Оператори
   - Интерпункциски знаци
   - Коментари
2. **Наброј ги основните елементи на програмскиот јазик C++:**
   - Податоци
   - Константи
   - Променливи
   - Декларации
   - Изрази
   - Искази

- Функции
- Модули

3. **Што се резервирани зборови? Наброј некои од нив:**
   - Веќе дефинирани зборови со специјално значење
   - Примери: int, if, else, while, return
4. **Што се идентификатори? Како се креираат?**
   - Имиња за променливи, функции и сл. што ги дава програмерот
   - Почнуваат со буква или _, без специјални знаци, не смеат да бидат резервирани зборови
5. **Примери на точно креирани идентификатори:**
   - zbir, broj1, _vrednost, tempVar
6. **Што се функции? Која функција мора да ја има програмата?**
   - Групи на наредби кои извршуваат одредена задача
   - main() е задолжителната главна функција

7. Што се коментари? Пример:

- Текст кој не се извршува, само објаснува
- Пример: // Ова е коментар

4.6 Искази во C++

Што се искази?

Исказите се **наредби** што му кажуваат на компјутерот што да направи. Без нив, програмата не би функционирала. Тие се извршуваат **по редоследот по кој се напишани**, еден по еден.

Секој исказ **завршува со точка и запирка (;)**.

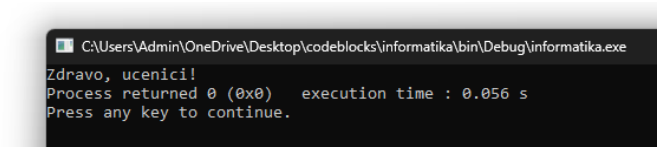
4.6.1 Исказ за приказ на екран

Во C++ за да прикажеме текст или број на екран, користиме cout.

🔹 Пример:

```cpp
#include <iostream>
using namespace std;

int main() {
    cout << "Zdravo, ucenici!";
    return 0;
}
```

🔹 Овој код ќе прикаже:



🔹 Структура на cout исказ

Дел	Објаснување
cout	Наредба за приказ на екран
<<	Оператор што испраќа содржина кон екранот
" "	Наводници што ја опкружуваат пораката
;	Крај на исказот

🔹 Употреба на endl

Ако сакаме да прикажеме повеќе редови, го користиме endl за нова линија:

```cpp
#include <iostream>
using namespace std;

int main() {
    cout << "Prv red" << endl;
    cout << "Vtor red";
    return 0;
}
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\informatika\bin\Debug\informatika.exe
Prv red
Vtor red
Process returned 0 (0x0)   execution time : 0.053 s
Press any key to continue.
```

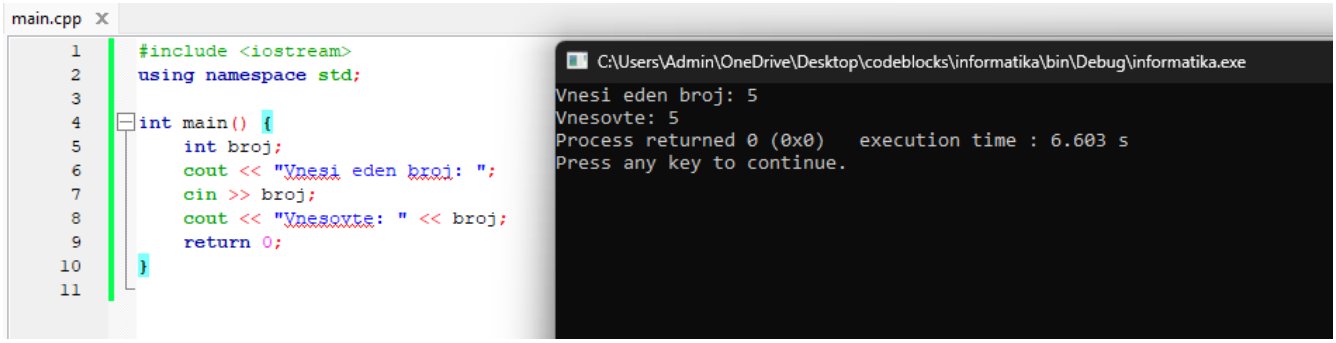
🔹 Секвенца од искази

➡ Кога имаме повеќе cout искази напишани еден под друг, тие се извршуваат **редоследно** – ова се нарекува **секвенца од искази**.

4.6.2 Исказ за доделување вредности

За да внесеме вредност од тастатура во програмата, користиме cin.

Пример:



Структура на cin исказ

Дел	Објаснување
cin	Внес на податок од тастатура
>>	Оператор за внес
broj	Променлива во која се чува внесениот број

Што се променливи?

Променливи се именувани простори каде што се **чуваат вредности**. Во примерот погоре, broj е променлива од тип int (цел број).

Променливите мора да се **дефинираат пред да се користат**, на пример:

cpp
CopyEdit
int broj; // дефиниција

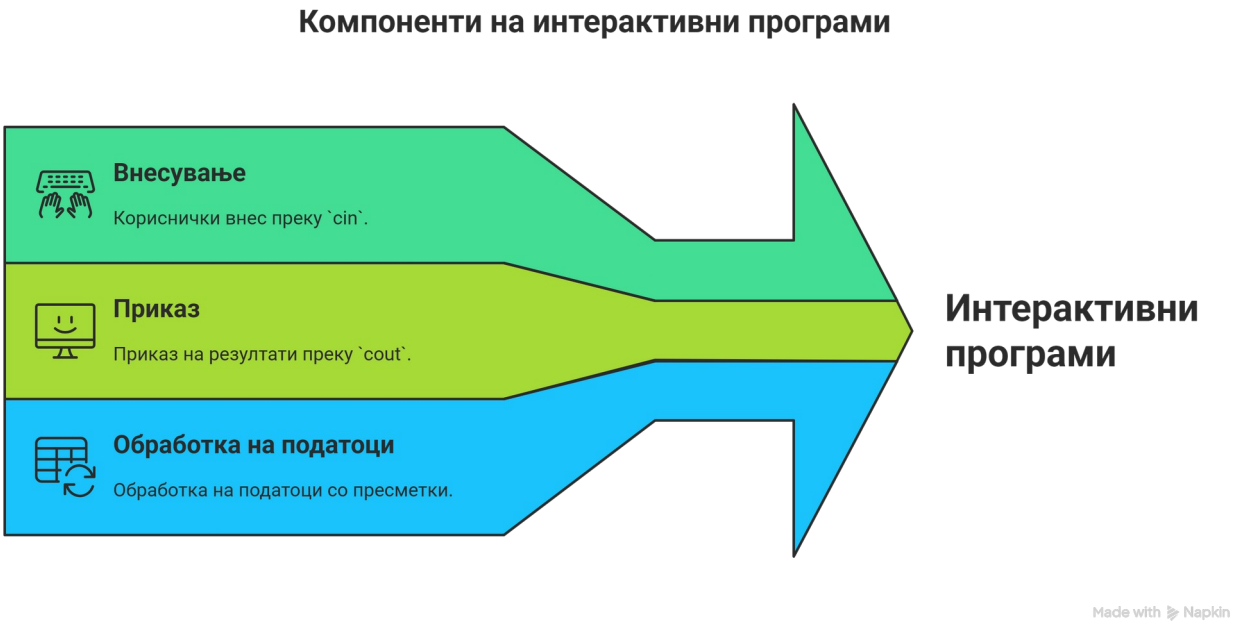
Доделување вредност со пресметка

Можеме да доделиме вредност на променлива преку **внес** или **пресметка**:

cpp
CopyEdit
int br1, br2, zbir;
cin >> br1 >> br2;
zbir = br1 + br2;
cout << "Zbir: " << zbir;

Интерактивна програма

Програма која **комуницира со корисникот** преку тастатура и прикажува резултат е наречена **интерактивна програма**.



Запомни!

- **Искази** се инструкции кои кажуваат што да прави програмата.
- Секој исказ завршува со ;
- cout → приказ на екран
- cin → внес на вредност
- endl → нов ред
- Променлива може да добие вредност преку **внес од тастатура** или **пресметка**

Прашања за проверка:

1. Што претставува еден исказ?
2. Кој C++ исказ се користи за прикажување текст?
3. Како се менува ред со cout?

4. Кој е C++ исказот за внесување вредност?
5. Што е интерактивна програма?

4.7 Изработка на програми

Во ова поглавје ќе учиме **како да пишуваме цели програми**, користејќи повеќе cout искази кои се **редоследно извршуваат**. Овој начин на извршување се нарекува:

 Техника на редоследно извршување
 Исказите се извршуваат еден по еден, по редот по кој се напишани.

▶ Пример 1: Испис на триаголник

Во оваа програма, ќе користиме cout за да отпечатиме **триаголник** составен од ѕвездички *.

Изворен код:

main.cpp

```cpp
#include <iostream>
using namespace std;

int main() {
    cout << "*" << endl;
    cout << "**" << endl;
    cout << "***" << endl;
    cout << "****" << endl;
    cout << "*****" << endl;
    return 0;
}
```

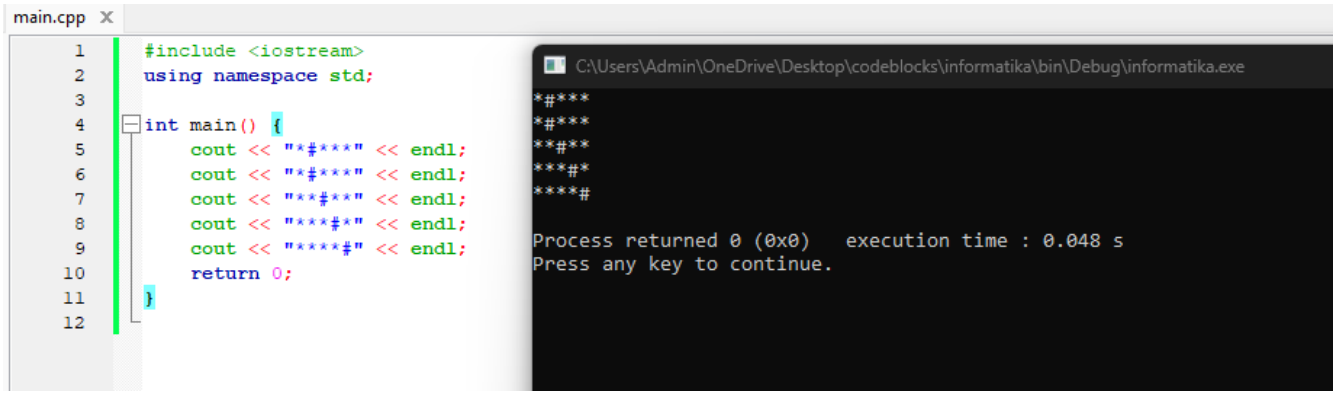
```
C:\Users\Admin\OneDrive\Desktop\codeblocks\informatika\bin\Debug\informatika.exe
*
**
***
****
*****

Process returned 0 (0x0)   execution time : 0.050 s
Press any key to continue.
```

 Секој cout испишува една линија, а endl ја преместува пораката во нов ред.

Пример 2: Квадрат со дијагонала

Во оваа програма ќе испишеме **квадрат**, но со дијагонала направена од друг симбол – на пример, #.

Изворен код:

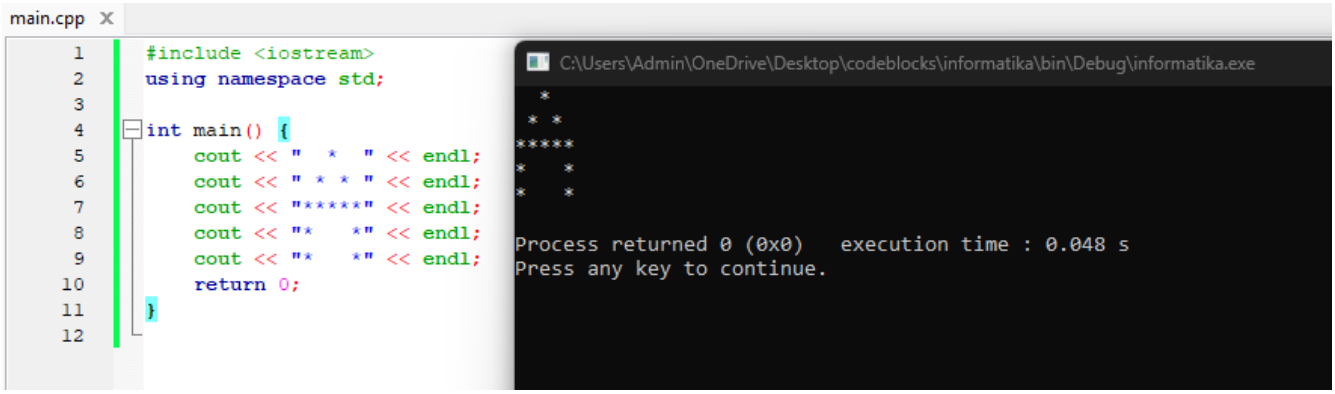


 Овде се комбинираат ѕвездички и # за да се создаде визуелна форма со **дијагонала**.

✍ Пример 3: Буква од твоето име

Ајде да бидеме креативни!
Можеш да ја испишеш **првата буква од твоето име**, срце ❤, или друга форма само со cout.

Пример – Буква А:



СОВЕТ:

Користи различни **симболи** и **простор (space)** за да креираш уметнички форми преку cout.
 Можеш да користиш #, *, @, o, + и слично.

Резиме

 Редоследно извршување значи:

- Секој исказ се извршува **еден по еден**
- Секој cout ја печати својата линија

- endl значи нов ред

 Со оваа техника, можеш да **исцрташ форми** само со текст!

🧠 Запомни!

- Програмите се пишуваат од **повеќе искази**
- Секој cout приказ завршува со ;
- **Техниката на редоследно извршување** е основа на секоја програма
- Можеш да креираш **визуелни форми** со симболи и cout

🧠 Прашања за проверка:

1. Што претставува техника на редоследно извршување?
2. Што ќе прикаже програмата ако напишеш 3 cout искази по ред?
3. Како можеш да направиш дијагонала во квадрат со cout?
4. Кој симбол се користи за нов ред?
5. Пробај сам да напишеш програма што печати срце! ❤

4.8 Аритметички операции и изрази

Кога пишуваме програми на C++, многу често ќе треба да правиме **математички пресметки**. За таа цел, ги користиме:

- **Аритметички изрази** – комбинации од броеви и математички оператори

Што е аритметички израз?

Аритметички израз е комбинација од **два или повеќе броеви** поврзани со математички оператори, кои го даваат резултатот од некоја пресметка.

Основни аритметички оператори во C++

Оператор	Опис	Пример во C++
+	собирање	zbir = a + b;
-	одземање	razlika = a - b;
*	множење	proizvod = a * b;
/	делење	kolicnik = a / b;

⚠ Во C++ **нема** оператор за експонентирање (^). За тоа се користи функцијата pow() од библиотеката cmath.

▶ Пример: Пресметки со два броја

📄 Изворен код:

main.cpp

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main() {
    int br1, br2;
    int zbir, razlika, proizvod;
    float kolicnik;

    cout << "Vnesi dva broja: ";
    cin >> br1 >> br2;

    zbir = br1 + br2;
    razlika = br1 - br2;
    proizvod = br1 * br2;
    kolicnik = float(br1) / br2;

    cout << "Zbir: " << zbir << endl;
    cout << "Razlika: " << razlika << endl;
    cout << "Proizvod: " << proizvod << endl;
    cout << "Kolicnik: " << kolicnik << endl;

    return 0;
}
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\informatika\bin\Debug\informatika.exe
Vnesi dva broja: 3 2
Zbir: 5
Razlika: 1
Proizvod: 6
Kolicnik: 1.5

Process returned 0 (0x0)   execution time : 6.150 s
Press any key to continue.
```

📌 Забелешка: експонентирање

Во C++ за **степенување**, се користи:

```
#include <cmath>
pow(broj, stepen);
```

Пример:

```
int rezultat = pow(2, 3); // 2 на трета = 8
```

📝 Резиме

📌 Аритметичките изрази се дел од речиси секоја програма.
📌 Прво внесуваме броеви со cin, па правиме пресметки со оператори.
📌 Резултатите ги испишуваме со cout.

📌 Запомни!

- Аритметички оператори: +, -, *, /
- За степенување користи pow() од cmath
- Делбата секогаш дава точен резултат ако користиме float

📝 Прашања:

1. Кои се основните аритметички оператори во C++?

2. Кој оператор се користи за множење?
3. Како се внесуваат вредности од тастатура?
4. Што се добива ако поделиме 5 на 2 во целобројна променлива?
5. Кој е резултатот на pow(3, 2)?

4.9 Константи и променливи

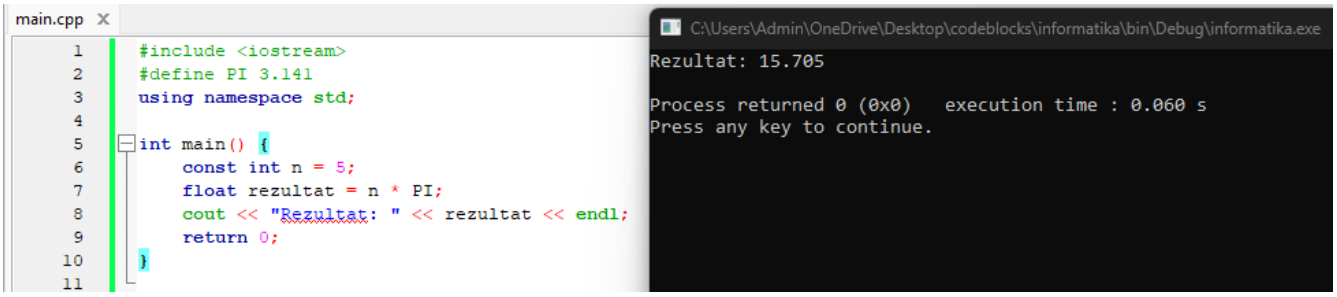
4.9.1 Константи

Константа е вредност која **не се менува** додека се извршува програмата.
На пример: pi = 3.141 е број кој никогаш не се менува.

Како се дефинира константа во C++?

Метод	Синтакса	Забелешка
Со #define	#define pi 3.141	Се пишува **над** main() и нема ;
Со const	const int n = 5;	Се пишува **во** main() и има ;

Пример со двете константи:



4.9.2 Променливи

Променливите се вредности кои **можат да се менуваат** во текот на извршување на програмата. На пример, бројот што корисникот ќе го внесе преку тастатура.

4.9.2.1 Тип на променлива

Секогаш мора да го наведеме **типот** на податокот што го чува променливата:

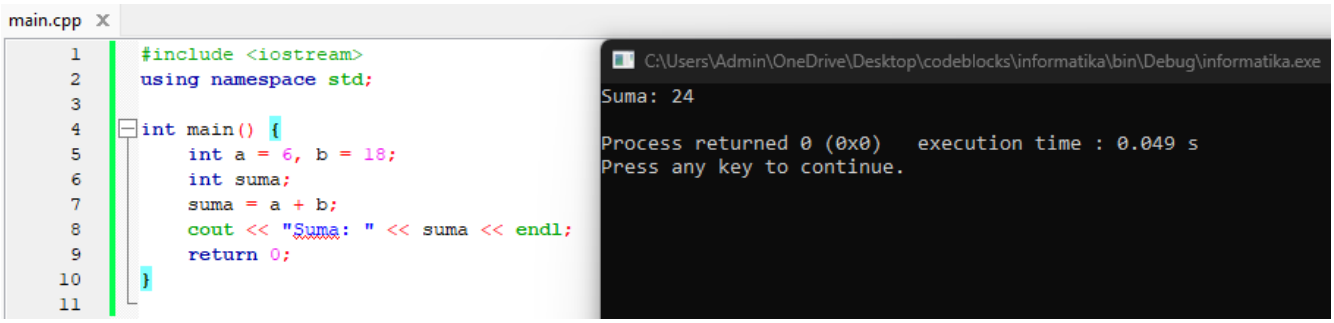
Тип	Опис	Пример
int	цел број	int broj = 5;
float	децимален број	float a = 2.78;
char	еден знак	char bukva = 'A';
bool	логичка вредност (0/1)	bool eTocno = true;

4.9.2.2 Додавање вредност (иницијализација)

Иницијализација = доделување почетна вредност на променлива:

```
int a = 5;
float b = 2.5;
int c;
c = a + 3;
```

Пример:



⚖ Разлика меѓу константа и променлива

Карактеристика	Константа	Променлива
Вредност	Секогаш иста	Може да се менува
Дефинирање	const, #define	int, float и сл.
Пример	const int d = 10;	int x = 7;

Запомни!

- **Константа**: вредност што не се менува.
- **Променлива**: вредност што може да се менува.
- Прво мора да се **декларира** типот и името на променливата.
- Почетната вредност се доделува со операторот = (**иницијализација**).

Прашања за проверка:

1. Што е константа, а што променлива? Објасни со свои зборови.
2. Која е разликата меѓу const и #define?
3. Може ли да се декларираат повеќе променливи одеднаш? Пример?
4. Што е иницијализација? Дај пример!
5. Кој тип би го користел за децимален број? А за еден знак?

4.10 Искази (техники) за внесување податоци во програмата

Што ќе научиме тука?

Во овој дел ќе научиме **како корисникот може да внесе податоци во програмата**. На пример, ќе направиме програма што бара од корисникот да внесе број, а потоа прави пресметка со тој број.

Техника на внесување податоци

За внесување податоци од тастатура во C++ се користи објектот cin и операторот >>.

```
cin >> promenliva;
```

Прво мора да ја **дефинираме променливата** што ќе ја користиме.
Потоа, со cin >> imeNaPromenlivata, му кажуваме на компјутерот **што да побара од корисникот да внесе**.
Со ова ја постигнуваме интеракцијата помеѓу човек и компјутер – го правиме програмот „паметен“ и флексибилен.

Задача 1: Пресметка на квадрат на внесен број

Објаснување:

Оваа задача ќе нè научи **како да внесеме еден број преку тастатурата**, и како потоа **да пресметаме квадрат** од тој број. Квадрат значи: број × број.

На пример: ако внесеме 4 → 4 × 4 = 16.

Решение:

main.cpp

```cpp
#include <iostream>
using namespace std;

int main() {
    int broj;
    cout << "Vnesi eden broj: ";
    cin >> broj;

    int kvadrat = broj * broj;

    cout << "Kvadratot na brojot e: " << kvadrat << endl;
    return 0;
}
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\Informatika\bin\Debug\informatika.exe
Vnesi eden broj: 3
Kvadratot na brojot e: 9

Process returned 0 (0x0)   execution time : 7.996 s
Press any key to continue.
```

🧮 Задача 2: Пресметка на плоштина и периметар на круг

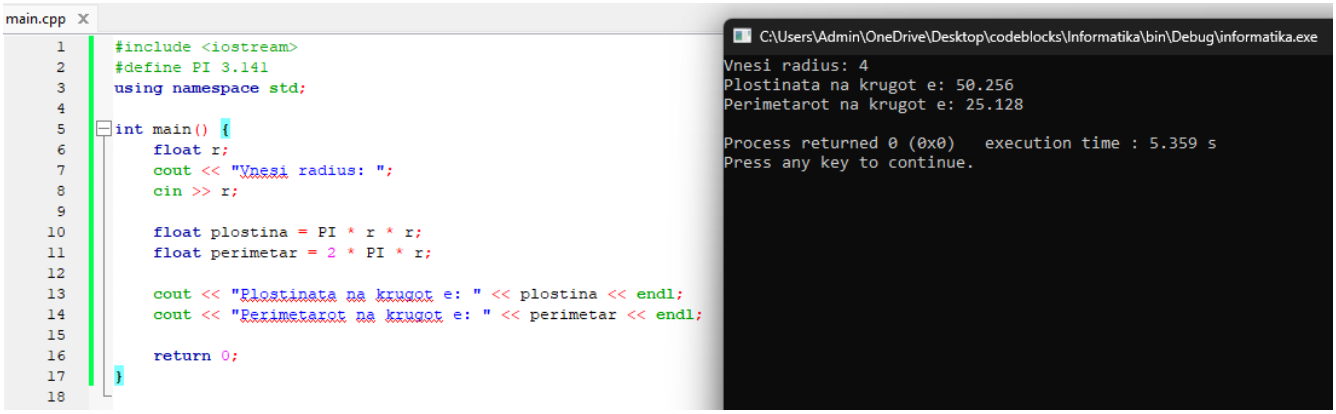
Објаснување:

Во оваа задача ќе научиме да **внесеме радиус на круг**, а потоа да ја пресметаме **плоштината и периметарот**. Ќе користиме две математички формули:

- Плоштина: P = π * r * r
- Периметар: L = 2 * π * r

Променливите ќе бидат од тип float, бидејќи очекуваме децимални вредности (π = 3.141).

💻 Решение:



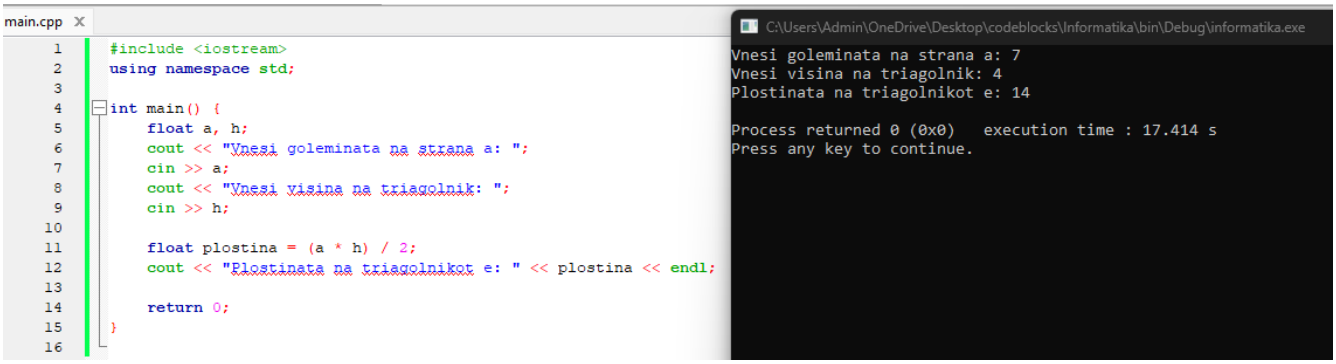
🧮 Задача 3: Пресметка на плоштина на триаголник

Објаснување:

Во оваа задача ќе внесеме **должина на основа и висина на триаголник**, и ќе ја пресметаме плоштината. Се користи математичката формула:

- Плоштина: P = (a * h) / 2

💻 Решение:



Задача 4: Претходник и следбеник

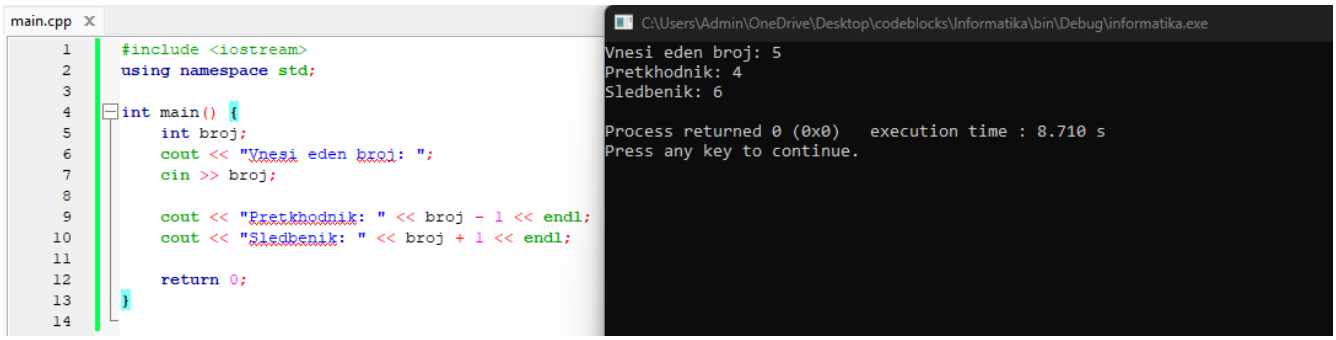
Објаснување:

Тука ќе внесеме еден број, а потоа ќе ги испишеме:

- бројот пред него (претходник)
- бројот после него (следбеник)

Оваа задача е лесна, но покажува **употреба на внес, пресметка и приказ**.

Решение:



Повтори и запомни:

Поим	Објаснување
cin	Објект за внес на податоци од тастатура
>>	Оператор за внесување
cout	Објект за приказ на податоци
<<	Оператор за печатење
promenliva	Променлива што ја користиме во програмата

Поим	Објаснување
=	Оператор за доделување вредност

🧠 Прашања за вежбање:

1. Како се внесува податок во C++?
2. Кој оператор се користи со cin?
3. Како се пресметува квадрат на број во програмата?
4. Која е формулата за пресметка на плоштина на круг?
5. Што се случува ако внесеме децимален број во int променлива?

4.11 Споредбени изрази

Во секојдневниот живот, често сме соочени со ситуации во кои треба да донесеме одлука – како на пример:

- 🤔 Дали да продолжиме да учиме или да почнеме да работиме?
- 🤔 Дали еден пат е подобар од друг?
- 🤔 Дали нешто вреди или не?

Овие одлуки се базираат на **споредби**:

- Дали едната опција е подобра од другата?
- Дали нешто е доволно добро или не?

На сличен начин и **компјутерите** работат со споредбени изрази – кога треба да се донесе одлука во текот на извршувањето на програмата. На пример, ако корисникот внесе број – програмата може да одлучи:

- Дали бројот е позитивен или негативен?
- Дали е делив со 2?

- Дали е поголем од 100?

Овие одлуки се прават преку **споредбени изрази**, кои во програмскиот јазик C++ се нарекуваат **логички изрази (Boolean expressions)**.

Што е споредбен израз?

Споредбен израз е услов кој може да биде:

- точен (true) или
- неточен (false)

Ако условот е **точен**, програмата продолжува со една акција.
Ако условот е **неточен**, таа може да избере друга насока.

Споредбени оператори во C++

Во C++ ги користиме следниве **оператори за споредба**:

Оператор	Опис	Пример (a=5, b=7)
==	еднакво на	a == b → false
!=	не е еднакво на	a != b → true
>	поголемо од	b > a → true
<	помало од	a < b → true
>=	поголемо или еднакво на	a >= b → false
<=	помало или еднакво на	a <= b → true

Зошто се важни споредбените изрази?

Со помош на споредбените изрази, можеме да ја **промениме насоката** на извршувањето на програмата.
Ова се нарекува **разгранување**.

На пример:

- Ако ученик има повеќе од 50 поени → положил.
- Ако температурата е под 0 → прикажи порака за мраз.
- Ако корисникот е најавен → прикажи мени.

Процес на разгранување во програмирањето
Проценка на состојбата
Проверка на состојбата на променливата или условот
Споредбен израз
Споредување на состојбата со одредена вредност
Одлука за разгранување
Одлучување дали да се разграни или не
Извршување на разгранување
Извршување на специфичен блок код врз основа на одлуката
Made with Napkin

🔍 Практична примена

За да се користат овие изрази во програми, во C++ се користи наредбата if, а понекогаш и if...else.

Во следниот дел (4.11.1), ќе објасниме **структурата за избор од две можности**, и ќе покажеме примери каде:

- се проверува дали некој е полнолетен,
- дали број е парен или непарен,
- кој од два броја е поголем,
- дали број е позитивен, негативен или нула.

4.11.1 Структура за избор од две можности

Во програмирањето, често имаме ситуации каде што сакаме да провериме дали некој услов е **исполнет** – и врз основа на тоа, да извршиме соодветна акција.

Оваа техника се вика **избор од две можности**, и е основа на сите одлуки што ги носи една програма.

Едноставно разгранување: if

Најпрво, го користиме if кога сакаме да извршиме **една** наредба само ако е исполнет некој услов.

На пример:

```
if (godini <= 18)
   cout << "Maloleten ste!";
```

Важно!
По if **не се става** точка и запирка ;, бидејќи тоа би значело крај на изразот.
Наместо тоа, после условот **директно** се пишува наредбата (или блок од наредби) што треба да се изврши ако условот е точен.

Двократно разгранување: if...else

Но, што ако сакаме да направиме **една работа ако условот е точен**, а **друга ако е неточен**?

Тогаш ја користиме **структурата** if...else.

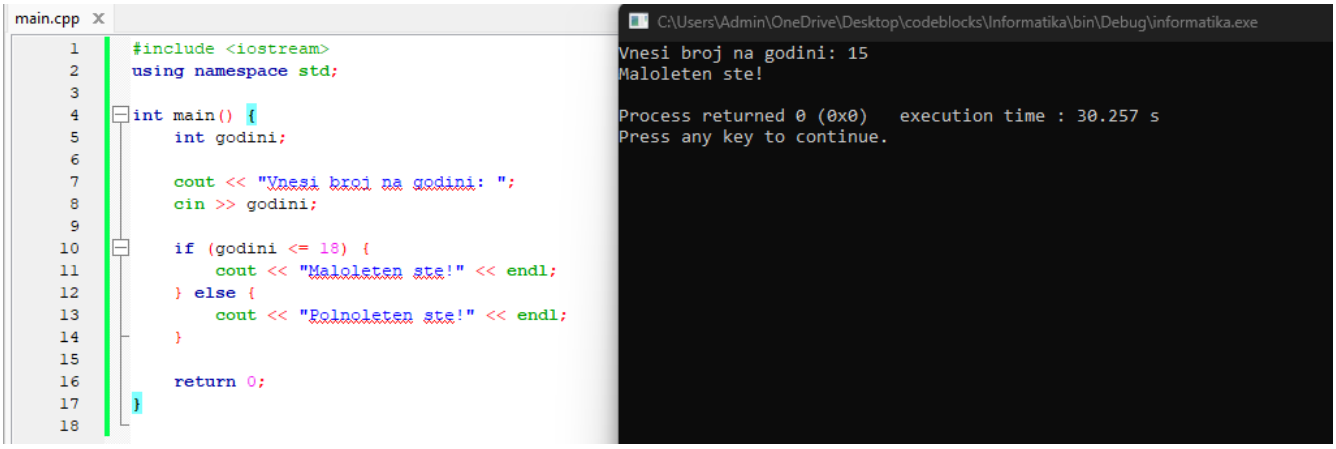
Пример:

Сакаме да провериме дали личноста е малолетна или полнолетна според внесени години.

Објаснување:

- Го внесуваме бројот на години.
- Ако бројот е **помал или еднаков на 18**, лицето е *малолетно*.
- Инаку (ако бројот е поголем), лицето е *полнолетно*.

Програма:



Блок од искази

Понекогаш сакаме да извршиме **повеќе наредби** ако условот е исполнет.

Во тој случај, ги ставаме во **блок од големи загради** {}, како во примерот подолу:

```
if (godini <= 18) {
   cout << "Maloleten ste!" << endl;
   cout << "Nemate pravo na glasanje." << endl;
} else {
   cout << "Polnoleten ste!" << endl;
   cout << "Mozete da glasate." << endl;
}
```

Што научивме тука?

- Со if проверуваме **дали еден услов е вистинит**.
- Со if...else можеме да избереме **една од две можности**.
- Со {} можеме да групираме повеќе наредби како **еден блок** што ќе се извршува заедно.

4.11.2 Изработка на програми со структура за избор од две можности

Во оваа секција ќе примениме споредбени изрази и условни искази (if, if...else) за да креираме програми кои проверуваат состојби и врз основа на нив извршуваат различни акции.

Задача 1: Да се пресмета плоштина на триаголник

Објаснување:
Прво ќе ја внесеме основата a и висината h на триаголникот. Потоа ќе ја пресметаме плоштината со формулата:

$P=(a \cdot h)/2$

Код:

main.cpp

```cpp
#include <iostream>
using namespace std;

int main() {
    float a, h, plostina;

    cout << "Vnesi golemina na strana (a): ";
    cin >> a;

    cout << "Vnesi visina (h): ";
    cin >> h;

    plostina = (a * h) / 2;

    cout << "Plostinata na triagolnikot e: " << plostina << endl;

    return 0;
}
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\Informatika\bin\Debug\informatika.exe
Vnesi golemina na strana (a): 5
Vnesi visina (h): 7
Plostinata na triagolnikot e: 17.5

Process returned 0 (0x0)   execution time : 5.286 s
Press any key to continue.
```

Задача 2: Проверка дали број е парен или непарен

Објаснување:
Ќе внесеме цел број и ќе провериме дали при делење со 2 дава остаток. Ако **остатокот е 0**, бројот е **парен**. Во спротивно, тој е **непарен**.

Код:

main.cpp

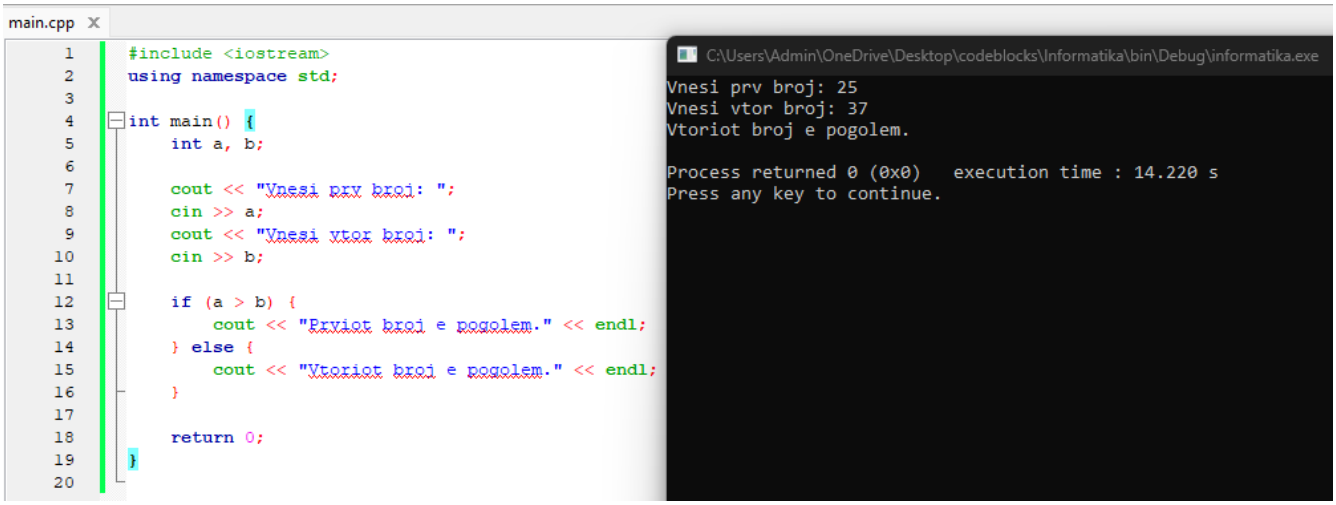
```cpp
#include <iostream>
using namespace std;

int main() {
    int broj;

    cout << "Vnesi eden broj: ";
    cin >> broj;

    if (broj % 2 == 0) {
        cout << "Brojot e paren!" << endl;
    } else {
        cout << "Brojot e neparen!" << endl;
    }

    return 0;
}
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\Informatika\bin\Debug\informatika.exe
Vnesi eden broj: 8
Brojot e paren!

Process returned 0 (0x0)   execution time : 6.269 s
Press any key to continue.
```

🔹 Задача 3: Да се провери кој број е поголем

Објаснување:
Внесуваме два броја. Ако првиот е поголем од вториот – тоа ќе се прикаже. Во спротивно, вториот е поголем.

Код:



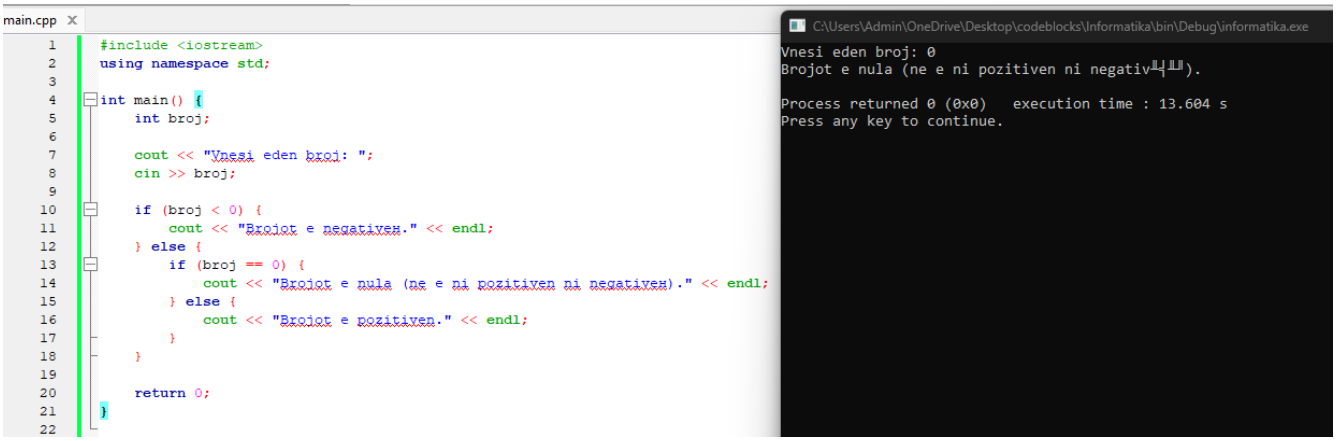
🔹 Задача 4: Проверка дали бројот е позитивен, негативен или нула

Објаснување:
Ќе внесеме број и ќе провериме:

- Дали е помал од 0 (негативен)
- Дали е еднаков на 0
- Дали е поголем од 0 (позитивен)

Овде користиме **вгнездени** if...else **изрази**.

Код:



🔁 Повторување и прашања

🔹 1. Дефинирај ги споредбените изрази!

Споредбените изрази се логички изрази кои враќаат вредност true (вистинито) или false (невистинито) врз основа на условна проверка.

🔹 2. Наведи ги операторите кои се употребуваат за споредба!

Оператор	Опис
==	еднаквост
!=	различно
>	поголемо
<	помало
>=	поголемо или еднакво
<=	помало или еднакво

🔹 3. Какво може да биде разгранувањето во програмата?

- **Еднократно** (if) – ако условот е точен, се извршува наредба.
- **Двократно** (if...else) – ако условот е точен, се извршува една наредба; инаку, друга.

🔹 4. Напиши ги синтаксите на If и If...Else!

```
// Еднократен избор:
if (uslov) {
   // наредби ако условот е точен
```

```
}

// Двократен избор:
if (uslov) {
   // наредби ако условот е точен
} else {
   // наредби ако условот не е точен
}
```

4.12 Структура за повторување во циклус до исполнување на услов

Во претходните лекции, видовме дека C++ исказите вообичаено се извршуваат редоследно – еден по еден. Овој начин се нарекува **физички редослед** на извршување. Но, во програмирањето често имаме потреба некој дел од кодот да се **повторува повеќепати**, сè додека е исполнет некој услов – тоа се нарекува **циклус** или **структура за повторување**.

Циклусот while во C++

Циклусот while е основна контролна структура која **проверува услов пред секое извршување**. Ако условот е вистинит (true), тогаш се извршуваат наредбите во телото на циклусот. Во спротивно, циклусот завршува.

Синтакса:

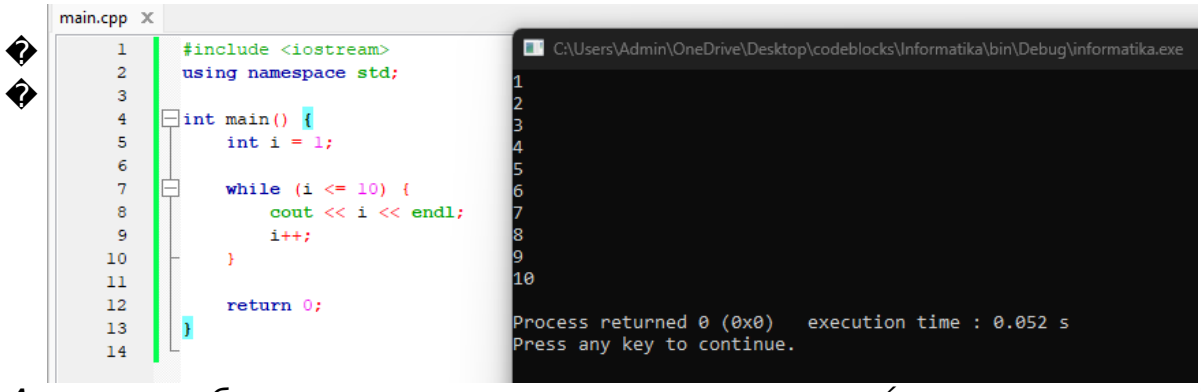
```
while (услов) {
   // наредби што се повторуваат
}
```

▶ Примери

🔹 Пример 1: Испишување броеви од 1 до 10

Објаснување:
Го дефинираме бројачот i со почетна вредност 1. Циклусот while ќе го испишува секој број сè додека i <= 10.

Код:

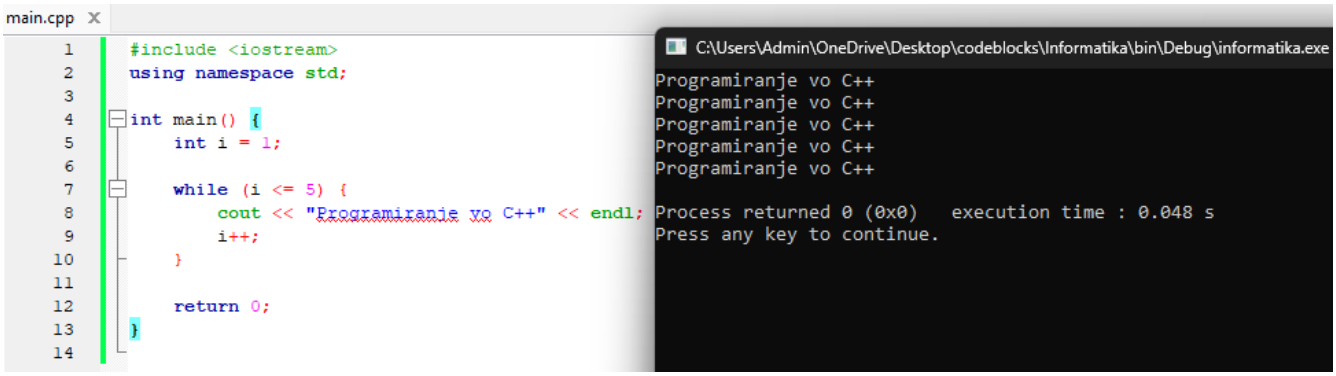


Ако сакаме броевите да се прикажат во еден ред, ќе го замениме endl *со запирка или празно место.*

🔹 Пример 2: Испишување текст 5 пати (бројач)

Објаснување:
Овој циклус ќе ја испише реченицата „Programiranje vo C++“ точно 5 пати. Бројачот i расте до 5.

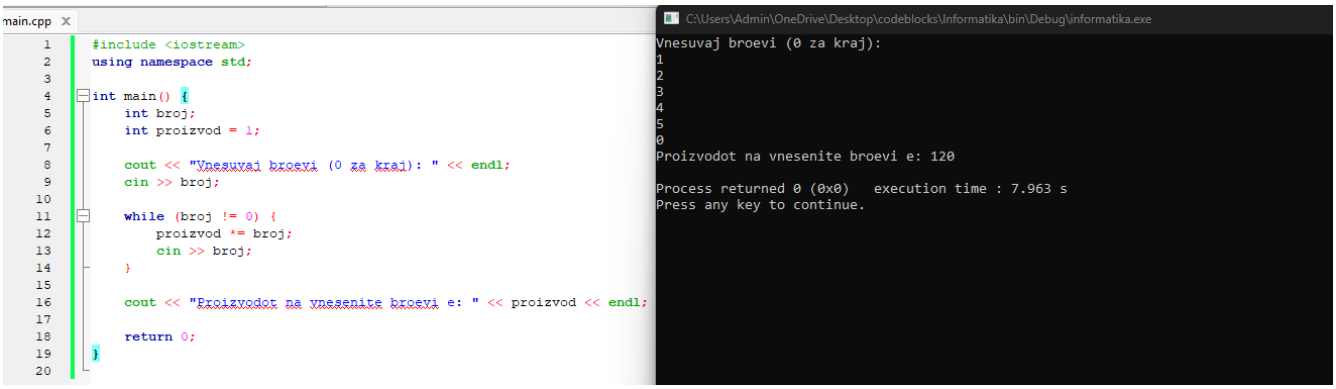
Код:



🔹 Пример 3: Производ на броеви додека не се внесе 0 (настан)

Објаснување:
Овој пример покажува циклус контролиран од **настан**, односно внес на број. Производот на броевите ќе се пресметува сè додека не внесеме 0, што го прекинува циклусот.

Код:



🔹 Повторување

🔹 **Кога се користи циклус** while**?**

- Кога **не знаеме однапред** колку пати треба да се повтори некоја акција.
- Кога условот треба да се **провери пред секое повторување**.

🔹 **Видови циклуси** while**:**

1. **Контролирани од бројач** – се користи променлива која се зголемува.
2. **Контролирани од настан** – се чека внес, одговор или услов што може да се промени во текот на извршувањето.

4.12.1 Изработка на едноставни програми со структура за повторување на циклус дури е исполнет услов

Во програмирањето, циклусите ни овозможуваат **повторување на наредби**, сè додека не се исполни одреден **услов**. Постојат две ситуации:

- 🔁 Кога **условот се проверува на почетокот** – со while циклус
- 🔁 Кога **условот се проверува на крајот** – со do-while циклус

Во овој дел ќе разгледаме примери на едноставни програми со **обе структури**.

▶ Пример 1: Условот се проверува на почетокот (while)

Задача: Пресметај го производот на последователни броеви до n. Ова значи 1 * 2 * 3 * ... * n.

main.cpp

```cpp
#include <iostream>
using namespace std;

int main() {
    int n;
    int proizvod = 1;
    int i = 1;

    cout << "Vnesi vrednost za n: ";
    cin >> n;

    while (i <= n) {
        proizvod *= i;
        i++;
    }

    cout << "Proizvodot e: " << proizvod << endl;

    return 0;
}
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\Informatika\bin\Debug\informatika.exe
Vnesi vrednost za n: 7
Proizvodot e: 5040

Process returned 0 (0x0)   execution time : 4.150 s
Press any key to continue.
```

🔍 *Овде, проверуваме дали* i <= n *пред да започне телото на циклусот.*

▶ Пример 2: Условот се проверува на крајот (do-while)

🔍 *Овде, телото на циклусот се извршува барем еднаш, па потоа се проверува условот.*

main.cpp

```cpp
#include <iostream>
using namespace std;

int main() {
    int n;
    int proizvod = 1;
    int i = 1;

    cout << "Vnesi vrednost za n: ";
    cin >> n;

    do {
        proizvod *= i;
        i++;
    } while (i <= n);

    cout << "Proizvodot e: " << proizvod << endl;

    return 0;
}
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\Informatika\bin\Debug\informatika.exe
Vnesi vrednost za n: 4
Proizvodot e: 24

Process returned 0 (0x0)   execution time : 20.674 s
Press any key to continue.
```

📝 Задачи за вежбање

🔹 Задача 1: Да се испишат парните броеви до n

main.cpp

```cpp
#include <iostream>
using namespace std;

int main() {
    int n;
    cout << "Vnesi broj n: ";
    cin >> n;

    int i = 1;
    while (i <= n) {
        if (i % 2 == 0)
            cout << i << " ";
        i++;
    }

    return 0;
}
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\Informatika\bin\Debug\informatika.exe
Vnesi broj n: 10
2 4 6 8 10
Process returned 0 (0x0)   execution time : 6.245 s
Press any key to continue.
```

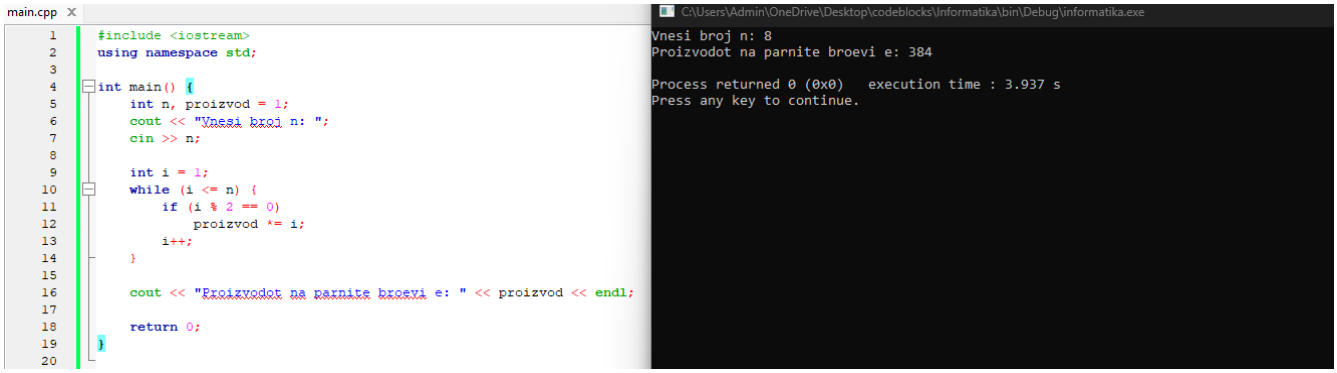
🔹 Задача 2: Да се пресмета збирот на непарни броеви до n

main.cpp

```cpp
#include <iostream>
using namespace std;

int main() {
    int n, zbir = 0;
    cout << "Vnesi broj n: ";
    cin >> n;

    int i = 1;
    while (i <= n) {
        if (i % 2 != 0)
            zbir += i;
        i++;
    }

    cout << "Zbirot na neparni broevi e: " << zbir << endl;

    return 0;
}
```

```
C:\Users\Admin\OneDrive\Desktop\codeblocks\Informatika\bin\Debug\informatika.exe
Vnesi broj n: 10
Zbirot na neparni broevi e: 25

Process returned 0 (0x0)   execution time : 5.326 s
Press any key to continue.
```

🔹 Задача 3: Да се пресмета производот на парните броеви до n



🔁 Повторување

🔁 Што се структури за повторување?

→ Тоа се делови од програмата кои се извршуваат повеќепати додека не се исполни одреден услов.

🔁 Најчесто користени циклуси се:

→ while, do-while, for

🔁 Синтакса на while:

```
while (услов) {
   // наредби
}
```

🔁 Синтакса на do-while:

```
do {
   // наредби
} while (услов);
```

🔁 Контрола:

- while – прво се проверува условот
- do-while – прво се извршуваат наредбите, потоа се проверува условот

🔁 Број на повторувања:

- Ако се знае однапред: користиме бројач (често со for)
- Ако не се знае однапред: користиме услов базиран на настан (while, do-while)

ДА ПОВТОРИМЕ! ДА УВЕЖБАМЕ!

🧠 Теоретски / Аналитички задачи

🔹 Задача A

Напиши програма со која ќе го добиеш следниов приказ на екран:

Prv cas: INFORMATIKA
Programiranje vo C++

🔹 Задача B

Кои се вредностите на променливите a **и** b**, по извршување на следните наредби?**

```
a = 15;
b = 23;
a = a + b;
b = a - b;
a = a - b;
```

🔹 Задача C

Што ќе се прикаже на екран по извршување на следниот код:

```
int a, b, c;
cin >> a;
cin >> b;
cin >> c;
cout << a << "," << b << "," << c << "." << endl;
```

Ако корисникот ги внесе вредностите: 5, 6, 7 (последователно)?

Задача D

Што ќе се испише на екран по извршување на следниот код?

```
int a = 8;
int b = 10;
cout << (a + b) / 2;
```

Задача E

Колку пати ќе се изврши следниот циклус?

```
int i = 2;
do {
   cout << i << " ";
   i++;
} while (i <= 10);
```

Практични задачи

Задача 1

Напиши програма за пресметка на плоштина на правоаголник.
Излезот на екран да ги прикаже димензиите на страните и резултатот.
Формула: P = a * b

Задача 2

Напиши програма која ќе пресметува аритметичка средина на два броја.
Како би се извршила истата пресметка за 8 броја?

Задача 3

Да се креира програма која проверува дали внесениот број е парен или непарен.

Задача 4

Да се креира програма која го пресметува периметарот на квадрат,
доколку внесената вредност за страната „a“ е позитивна.
Во спротивно, да се испише порака:
"Пресметката не може да се изврши! Внесете позитивен број!"

Задача 5

Со примена на циклусот do–while, да се пресмета збирот на парните броеви до n-ти број.

Задача 6

Да се пресмета збирот на броевите кои циклично се внесуваат од тастатура,
сè додека не се внесе бројот 0.

Вовед во веб дневници (блогови)

5. Вовед во веб-дневници (блогови)

🌐 Што е интернет?

Замислете огромна мрежа која ги поврзува сите компјутери во светот.
Таа се вика **Интернет**!
Преку него, можеме:

- да испраќаме пораки 📩
- да гледаме видеа 📺
- да слушаме музика 🎧
- да купуваме нешта 🛒
- да читаме интересни текстови 📖
- да пишуваме наши приказни или блогови ✍️

📝 Што е блог?

Блог (или веб-дневник) е место на интернет каде што можеш да споделуваш:

- твои мислења
- интересни приказни
- слики
- совети
- видеа

Замисли го како твој мал онлајн дневник – но наместо да го држиш во фиока, го читат и други луѓе! 🌍

🧑‍🤝‍🧑 Кој може да пишува блог?

Секој! Дури и ТИ!
Можеш да имаш блог за:

- твоето милениче 🐶
- храна што ја сакаш 🍕
- видеоигри 🎮
- патувања ✈️
- твоето секојдневие 🏫

🌟 Важно е да бидеме паметни корисници на интернет!

Интернет е супер, но мора да се **однесуваме безбедно и културно**!
Еве неколку правила:

🚫 НЕ споделувај лични податоци

- Име и презиме
- Адреса
- Телефонски број
- Фотографии
  🔍 Зошто? Бидејќи не знаеш кој е од другата страна на екранот!

💳 НЕ користи кредитни картички на родителите без дозвола!

🔐 Чувај ги твоите лозинки!

- Не ги кажувај на другарите
- Не ги пишувај на јавни места

🤝 Биди љубезен/на

- Не користи навредливи зборови
- Почитувај ги мислењата на другите
- Не пиши со ГОЛЕМИ букви – тоа значи дека викаш! 😮

Што значи дигитален отпечаток?

Твојот **дигитален отпечаток** е сè што оставиш зад себе на интернет – и добро и лошо.
Тоа може да биде нешто што **намерно** си објавил, но и нешто што се случило **случајно**.

🌟 Позитивен отпечаток

Кога го користиш интернетот **паметно**, твојот отпечаток може да биде **многу корисен**!

 Пример:

- Сакаш да научиш како да направиш пица
- Го пишуваш тоа во Google
- Ќе ти се појават блогови со рецепти и видеа
- Гледаш, учиш, готвиш – и сè тоа благодарение на интернетот!

 Тоа е позитивен дигитален отпечаток!

⚠ Негативен отпечаток

Но ако не внимаваш, интернетот може и да те „издаде“!

 Примери:

- Слика што мислиш дека е приватна – станува **јавна**
- Лични податоци – **незаштитени** и достапни за непознати
- Коментари или објави кои може некој да ги **злоупотреби**

 Секогаш прашај се пред да објавиш:

- Дали ова е паметна објава?
- Кој сè може да ја види?
- Дали ми носи корист или штета?

⚖ Како да одлучам дали нешто треба да објавам?

Замисли скала со две страни:

- ✔ Позитивни страни: дали ќе научи некој нешто? дали е корисно, забавно или инспиративно?
- Негативни страни: дали може некој да се потсмева? дали откривам премногу? дали би се срамел/а ако тоа го види наставник?

 Ако позитивното е посилно → објави!
 Ако негативното е посилно → размисли двапати!

🔐 Совети за безбеден и паметен отпечаток:

🔒 **Одјави се** од профилите кога завршуваш
🔒 **Не кажувај лозинки** дури ни на најдобар другар
🔒 **Не ги пишувај лозинките јавно**
🔒 **Користи јака лозинка** – со букви, броеви и знаци (!?#)
🔒 **Бриши редовно историја и кеш**
🔒 **Немој да делиш лични податоци** (адреса, училиште, телефон)

5.1 Поим за блог и негова примена

Блог = Веб-дневник

Замисли место на интернет каде што можеш да пишуваш твои мислења, приказни, објави, да споделуваш слики, видеа и да комуницираш со други луѓе.
➡ Тоа место се вика **БЛОГ**!

🔄 Секој блог е организиран по **временски редослед** – **најновите објави се прикажуваат најгоре**, а постарите се движат надолу.

🌐 Од каде потекнува зборот "блог"?

- **Web** = мрежа
- **Log** = запис или дневник

🧠 Значи, блог = **запис на интернет**!

🧑‍💻 Кој пишува блог?

Авторот на блогот се вика **блогер**!
Блогерот:

- 🆕 Го создава блогот
- ✏️ Ја уредува содржината
- ✍️ Пишува текстови
- 🔄 Го обновува редовно

Овој процес се нарекува **блогирање**.

 Секој што знае да користи компјутер и интернет, и сака да зборува или пишува за некоја тема – може да биде блогер!

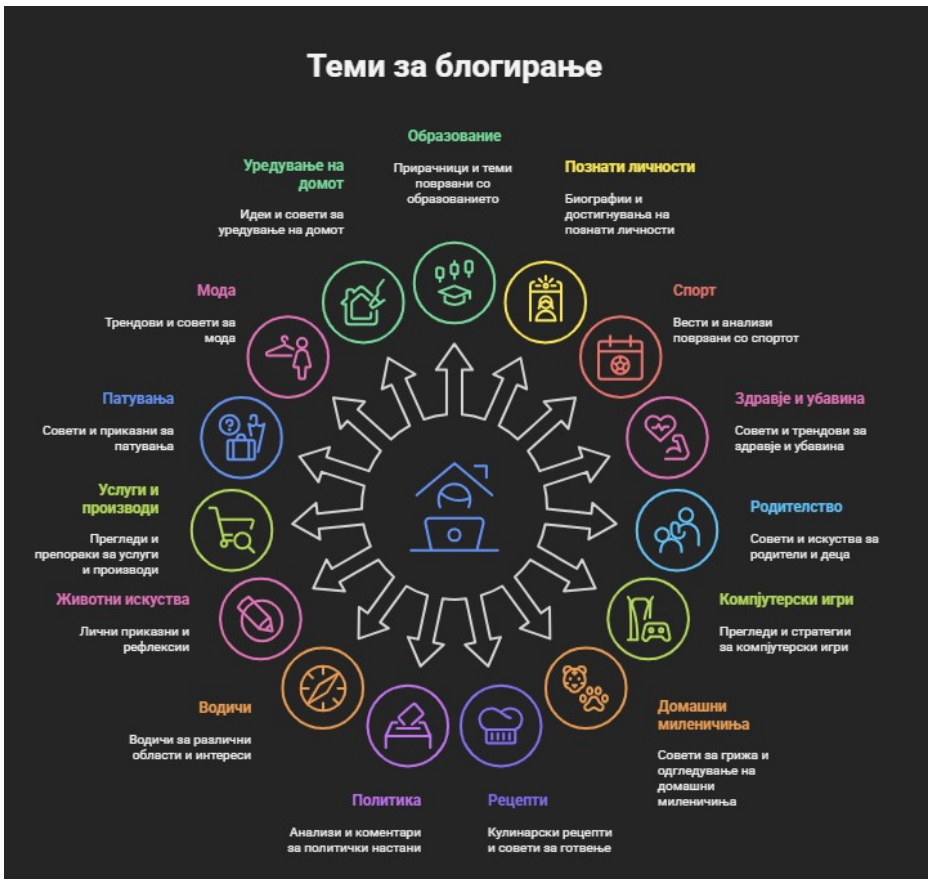
Што можеме да објавиме на еден блог?

Блогот може да содржи:

- Текстови и белешки
- Фотографии
- Видеа
- Линкови
- Коментари
- Прашања и дискусии

Тоа го прави блогот **динамичен и жив**, како мало интернет списание!

За кои теми може да се блогира?



Зошто луѓето блогираат?

 За да споделат знаење
 За да се поврзат со други што мислат исто
 За да коментираат важни теми
 За да си ја изразат креативноста
 За да учат од другите блогери

За кого е наменет блогот?

Блогот може да биде:

- Јавен – секој може да го чита
- Приватен – само одредена група, на пример: ученици во одделение

На пример:

- Наставник може да направи блог со задачи, видеа и вежби
- Ученици можат да коментираат, поставуваат прашања и разменуваат идеи

Вежба: Истражи блогови!

Користи интернет пребарувач (Google, Bing) и најди блогови за:

- Информатика
- Готвење
- Цртање и креативност
- Патувања
- Миленичиња

Обиди се да најдеш барем **еден блог** што е корисен за тебе!

5.2 Изработка на блог

Креирањето блог е многу лесно!

Ако знаеш да користиш интернет и Gmail – тогаш веќе си на половина пат до твојот прв блог!
За среќа, постојат многу **бесплатни веб-страници** кои ти дозволуваат да креираш свој блог **без пари**.

🌐 Најпознати места каде може да направиш блог:

🛠️ Име на сервис	🔎 Што е тоа?
Blogger	Најстар блог-сервис, дел од Google
WordPress	Многу популарен, лесен за користење
Tumblr	Креативен и визуелен блог
Medium	Супер за пишување мислења и статии
Wix / Weebly	Можеш да направиш и блог и веб-страница
Joomla / Drupal	Повеќе за напредни корисници
Ghost	Модерен блог-сервис за пишување
Squarespace / Typepad / Magento	Посериозни блогови и бизнис страници

🚀 Да зборуваме за Blogger!

🌟 Што е Blogger?

- Бесплатна платформа за блогирање
- Дел од Google – значи, **можеш да се најавиш со твоето Gmail корисничко име и лозинка**

- Твојата блог-адреса ќе изгледа вака:
  imenatvojotblog.blogspot.com

Што можеш да правиш на Blogger?

- Пишуваш статии и дневници
- Ставаш слики и видеа
- Објавуваш домашни проекти и трудови
- Правиш блог како весник или портал
- Дури можеш и да **заработуваш** (кога ќе пораснеш!) преку реклами

Немаш Gmail сметка?

Нема проблем! Еве како да почнеш:

Кога ќе отвориш www.blogger.com,
ќе видиш копче **"Sign In"** – тоа значи „Најави се“.
Ако **немаш Gmail сметка**, кликни таму и ќе те води чекор по чекор како да ја креираш.
(Потребна е само е-пошта, лозинка, и малку трпение!)

Ако **веќе имаш Gmail**, тогаш кликни **“Create New Blog”** –
ова значи „Креирај нов блог“ – и почнува магијата!

Внесување на информации за блогот

Кога ќе кликнеш на **“Create New Blog”**, ќе ти се појави **прозорец за најава**
(тука внесуваш твоето Gmail корисничко име и лозинка).

По најавата, ќе треба да пополниш **неколку важни работи:**

✍ 1. Име на блогот

Ова е насловот на твојот блог – пример:
„Моите игри и авантури“ или **„Готвиме со Ана“**

2. Адреса на блогот

 Ова е линкот преку кој другите ќе можат да го најдат твојот блог.
Пример: mojblog123.blogspot.com
 Системот ќе провери дали адресата е слободна →
ќе ти се појави порака: **"Checking Address Availability"**
Ако е зелено – супер, продолжуваме!

3. Избери тема (дизајн)

 Ова е изгледот на твојот блог – боите, распоредот, стилот.
Избери тема што **одговара на тоа за што ќе пишуваш**:
 модерна, шарена, класична, слатка...

Финален чекор: Create Blog

Штом ќе ги внесеш сите информации, кликни на копчето
 "Create Blog" – и твојот блог е спремен за старт!

Контролна табла (Dashboard)

Контролната табла е како твој „таен центар за блогирање“
Од таму можеш да:

- ✏ Напишеш нова објава (текст, слики, видеа)
- Измениш претходна објава
- Види коментари од читателите
- Организираш категории
- Смениш изгледот кога сакаш

Контролната табла изгледа **јасно и лесно за користење** – со копчиња за сè што ти треба.

Повтори и запомни:

Чекор	Што се прави?
1. Најава на Blogger	Со Gmail корисничко име и лозинка
2. Create New Blog	Креирање нов блог
3. Внесување	Име, адреса и избор на дизајн

🔢 Чекор	🧠 Што се прави?
податоци	
4. Проверка на адреса	Дали е слободна или веќе зафатена
5. Клик на “Create Blog”	Потврда и почеток на блогот
6. Dashboard	Место каде го уредуваш и менаџираш блогот

🎉 Блогот е како твој **дигитален дневник + списание + проект во едно**
Со само неколку клика – **можеш да се искажеш, споделиш и инспирираш!**

5.2.2 Уредување на содржина на блог

Штом се појави **контролната табла (Dashboard)** на вашиот блог, вие сте подготвени да започнете со вистинската магија – **креирање и уредување објави!**

📝 Како да започнеме нова објава?

1. Кликнете на копчето 📝 **New Post (Нова објава)**.
2. Ќе ви се прикаже прозорецот за уредување на нова објава:

✍ Што можете да направите тука?

Во овој едноставен едитор, можете да пишувате текст, да го форматирате и да вметнувате мултимедија – сè што ви треба за да направите интересна и привлечна објава!

Основни алатки:

- **Фонт:** Променете стил, боја, големина
- **Болд/Italic/Underline:** За нагласување на важни делови
- **Вметнување линк:** Поврзете го текстот со друга страница
- **Вметнување слика или видео:** Украсете ја вашата објава!

Поставки за објавата (Post Settings)

Од десната страна, ќе ги најдете **дополнителните опции**:

Опција	**Објаснување**
Labels (Ознаки)	Додадете клучни зборови (на пр. "Школо", "Технологија")
Schedule (Распоред)	Дали сакате да се објави веднаш или на одреден датум?
Location (Локација)	Додадете од каде е напишана објавата
Permalink (Линк)	Можете да изберете сопствена веб-адреса за објавата

Објавување на објавата

Кога ќе завршите со пишување:

- **Publish (Објави):** ја објавува веднаш
- **Save (Зачувај):** ја чува како нацрт
- **Preview (Преглед):** ви покажува како ќе изгледа
- **Close (Затвори):** го напушта едиторот без да се објави

Совети за поинтересни објави

Напиши краток, интересен наслов
Додади слика која ќе го долови темата
Стави линкови до други корисни страници
Заврши со прашање за да ги вклучиш читателите (на пр. „А вие што мислите?“)

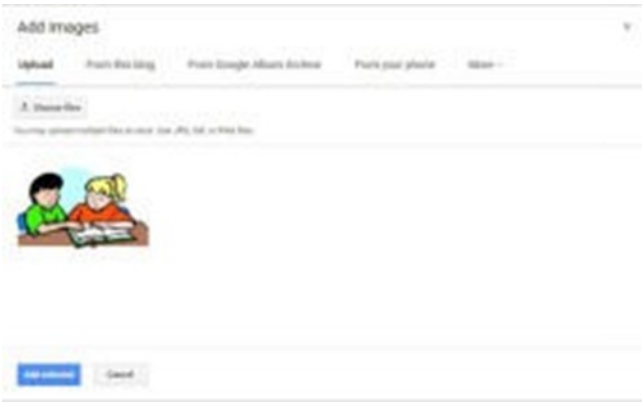
5.2.3 Додавање слика и видео во блог

Сликите и видеата се клучни за да направите објавата **повизуелна, поинтересна и полна со енергија!** Тие им помагаат на читателите побрзо да ја разберат пораката и да се задржат подолго на вашиот блог.

Како да додадете слика во блог објава?

1. Во прозорецот за креирање објава, поставете го **курсорот** на местото каде сакате да се појави сликата.
2. Потоа, кликнете на иконата за **вметнување слика**

Избор и прикачување на слика



- Кликнете на **Choose Files (Избери датотека)**
- Одберете слика од вашиот компјутер
- Потоа кликнете на **Add Selected (Додади избрана)**

 Совет: Користете сопствени фотографии или слики со лиценца за слободна употреба (на пример од pixabay.com или unsplash.com).

🛠️ Уредување на сликата

Откако ќе ја додадете сликата, кликнете на неа – ќе се појават **опции за уредување**:

⚙️ Опција	Што прави?
Small/Medium/Large/Original size	Промена на димензија
Left/Center/Right	Усогласување на сликата
Add caption	Додавање опис под сликата
Properties	Уреди наслов и ALT-текст
Remove	Бришење на сликата

Како да додадете видео?

Постапката за видеа е **многу слична**!

1. Кликнете на опцијата **Insert video (Вметни видео)** или **Choose video to upload (Избери видео за прикачување)**
2. Изберете видеото што веќе го имате зачувано на компјутер
3. Почекајте да заврши прикачувањето
4. Кликнете **Publish (Објави)**

(Слика 4: Прозорец за прикачување видео)

 Совет: Ако видеото ви е на YouTube, можете едноставно да го вметнете со copy-paste на линкот во објавата или со Embed код.

Креативен совет

 Размислете: **Како би изгледала објавата без слики или видеа?** Досадно, нели? Секогаш додавајте визуелни елементи за да го разиграте вашиот блог!

5.2.4 Додавање линкови во блог

Блогот не мора да биде само *текст*. Замислете го како **мрежа што ги поврзува вашите мисли со целиот интернет!** Со додавање линкови, можете да поврзете:

- веб-страници
- документи
- е-маил адреси
- видеа и други ресурси

Како да додадете линк?

1. Селектирајте го текстот кој сакате да го претворите во линк (на пр. „Кликни тука“).
2. Кликнете на иконата за **додавање линк** во алатките на едиторот:

🔗 Прозорец за уредување на линкот

Во него:

- **Text to display:** Текстот кој сте го селектирале
- Изберете:
  - 🌐 **Web Address** – за линк кон веб-страница
  - 📧 **Email Address** – за да го отвори е-маил клиентот
- Внесете ја адресата (URL или е-маил)
- Кликнете **OK** за потврда

💡 *Совет:* Секогаш проверувај дали линкот функционира правилно пред да објавиш!

🚀 Објавете ја објавата со клик на Publish

5.2.5 Менување на темата / изгледот на блогот

Дали вашиот блог изгледа досадно? ⬇
Сакате да биде шарен, модерен или класичен? ⬆
Со Blogger, **можете да го промените изгледот за само неколку клика!**

🎨 Како се менува темата?

1. Отворете ја **контролната табла (Dashboard)**
2. Кликнете на опцијата **Theme (Тема)**

Што потоа?

- Ќе се прикаже преглед како ќе изгледа вашиот блог
- Ако ви се допаѓа, кликнете на **Add (Додади)**
- За да видите како изгледа вашиот блог во живо – изберете **View Blog**

Бонус: Некои теми нудат опции за прилагодување на боите, фонтот и распоредот на елементите.

Да не заборавиме!

Blogger е еден од најчесто користените блог-сервиси
За да користите Blogger, треба да имате **Gmail сметка**
Преку **контролната табла (Dashboard)** можете да:

- креирате и уредувате објави,
- додавате слики, видеа и линкови,
- менувате изглед и тема на блогот,
- следите посетеност.

ЗАДАЧА: Ваш прв блог-пост! ✍

Цел: Да создадете прв блог пост кој ќе вклучува:

- Барем една **слика**
- Еден **линк до веб-страница**
- Кратко **видео или линк до YouTube видео**
- Да го смените изгледот на блогот со нова тема

Кога ќе завршите:

- Направете **преглед** на објавата

- Објавете го блогот!
- Поканете другар или родител да го погледне

5.3. Коментирање содржини на блоговите

Блоговите не се само платформи за читање – тие се **живи заедници** каде што сите можеме да се вклучиме, дискутираме и споделуваме!

 Дали имаш идеја?
 Сакаш да поставиш прашање?
 Или да кажеш „браво“ на нечиј пост?

Тогаш, време е да коментираш!

Зошто коментираме?

 Да споделиме искуства
 Да прашаме или одговориме
 Да дадеме поддршка
 Да предложиме решение
 Да започнеме дискусија

 Коментарите се алатка за вклучување во заедницата.

Како да оставиш коментар?

Процес на оставање коментар на блог
Отвори го блогот
Внеси ја адресата на блогот во прелистувачот
Најди објава што те интересира
Прелистај ја содржината
Кликни на копчето за коментар
Избери ја опцијата за коментар под објавата
Внеси го твојот коментар во полето за текст
Напиши коментар
Избери профил
Избери профил за коментарот
Избери дали да објавиш или да прегледаш
Избери опција за објавување

🛠️ Улога на администраторот

Администраторот може:

🔹 Да ги прегледа коментарите:

- Преку **Dashboard** > **Comments**

🔹 Да одговори на коментар:

- Кликнува на **Reply**
- Внесува одговор
- Кликнува **Publish**

🔔 Запомни!

🔹 **И корисниците и администраторот** можат да коментираат.
🔹 Коментарите се срцето на интеракцијата во блогот.
🔹 Преку нив учиме, дискутираме и се поврзуваме.

🎯 Задача за тебе!

🔹 **Избери една блог објава (реална или замислена)** и:

1. Напиши **примерен коментар** што поставува прашање.
2. Напиши **одговор како администратор** на тој коментар.

💡 Совети:

- Коментарот нека биде културен и поврзан со темата.
- Одговорот треба да биде поддржувачки и јасен.

АЈДЕ ДА ПОВТОРИМЕ!

Заокружи го точниот одговор

1. Луѓето кои ги уредуваат веб-дневниците се нарекуваат ➡
□ блогери
□ веб-дизајнери
□ програмери

2. Постапката на уредување на блоговите се нарекува ➡
□ програмирање
□ блогирање
□ дизајнирање

3.Како најчесто се организираат записите на блогот?
а) според бројот на сликите објавени
б) без некој посебен редослед
□ в) според хронолошки редослед, односно по датумот на објавата

□ *Зошто мислиш така? Објасни со свои зборови.*

4.Што е потребно пред да започнеш со изработка на блог?
а) □ неопходно е корисникот да има е-маил адреса
б) не мора да има е-маил
в) не е потребна е-маил адреса

□ *Зошто мислиш дека е важно да имаш е-маил адреса?*

5.Корисничкото име на авторот на блогот се користи за...
а) име за потпис
 б) име за најава во веб-дневникот
в) тема за дискусија

Замисли дека имаш свој блог – како би се викаш?

6.Записите на блогот се нарекуваат:
а) статуси
 б) постови
в) твитови

 Зошто не се нарекуваат твитови? Дали знаеш на која платформа се користи тој поим?

7.Посетителот на блогот може да остави порака преку опцијата за додавање:
а) линк
б) блог-архива
 в) коментари

Пробај да напишеш еден позитивен коментар како да си читател на нечиј блог!

8.Што е потребно да направиш за да се објави ново напишан запис на блогот?
 Кликни на копчето **„Објави“ (Publish)**

Во некои платформи, можеш и да закажеш објава за одреден датум! Зошто тоа е корисно?

ЗАДАЧА

Твоја мини-мисија:

1. Замисли дека имаш блог за твоето омилено хоби.
2. Напиши еден краток **наслов** на твојот прв пост.
3. Напиши 3-4 реченици што би стоеле во тој пост.
4. Додади еден замислен **коментар** од твој читател.

Можеш да го направиш ова во тетратка, или ако работите дигитално – во документ или PowerPoint.

РЕЧНИК НА КЛУЧНИ ПОИМИ

Компјутерски и програмски концепти

Карактеристика	Опис
Активна ќелија	Подготвена за внесување податоци
Анализа	Делење на проблемот на делови
Бинарни броеви	Јазикот на компјутерот од 0 и 1
Блог	Онлајн дневник за споделување мисли
Графикон	Визуелна репрезентација на резултатите од податоците
Графички приказ	Слика која ефективно пренесува информации
Дебагер	Алатка за наоѓање и исправка на грешки
Дигитален отпечаток	Траги оставени при сурфање на интернет
Едитор	Нотесник за пишување и уредување код
Елминирање	Отстранување на неважни ставки за наоѓање решение
Запис	Кратка белешка или објава на блог
Изворен програм	Листа на команди за компјутерот
Извршен програм	Програм подготвена за употреба по компилација
Иницијализација	Поставување почетна вредност за тајмер
Интерактивни програми	Програми кои одговараат на кориснички влез
Изјави (Команди)	Упатства дадени на компјутерот
Кодирање	Преведување идеи во компјутерски јазик
Колона	Вертикални линии во табела
Коментар	Објаснување на кодот за идни референци
Компилатор	Преведувач кој конвертира код помеѓу јазици

Made with Napkin

Аналогија на програмски концепти

Концепт	Аналогија
Константи	Непроменливи правила
Координати	Адреси во табелата
Криптографија	Шифрирање на пораки
Логичко размислување	Бити детектив
Настан	Притискање на копче
Непоследователни ќелии	Острови во табелата
Последователни ќелии	Вагоните во возот
Програмирање	Пишување упатства за робот
Променливи	Кутии за различни работи
Работна тетратка	Дебела тетратка со многу листови
Работен лист	Една страница од работната тетратка
Ред	Редови на седишта во кино
Нарачка	Листа на задачи
Бесплатен софтвер	Бесплатна игра
Булева експресија	Прашање "дали ова е точно?
Структури на податоци	Кутии за сортирање играчки
Табела	Голема мрежа со редови/колони
ќелија	Пресек на колона и ред
Формули	Математичка магија
Функции	Готови рецепти
Циклус	Песна на повторување

Made with Napkin

Книгата можете да ја преземете на овој линк: